A controlled study of the contribution of external resources on Chinese word segmentation

Yongshun Chen
B.E., Beijing University of Technology, 2004

Presented to the Center for Spoken Language Understanding within
the Department of Biomedical Engineering
and the Oregon Health & Sciences University
School of Medicine
in partial fulfillment of
the requirements for the degree of
Master of Science
in
Computer Science and Engineering

February 2011

The thesis “A controlled study of the contribution of external resources on Chinese word segmentation” by Yongshun Chen has been examined and approved by the following Examination Committee:

Brian Roark
Associate Professor
Thesis Research Advisor

Peter Heeman
Associate Research Professor

Aaron M. Cohen
Associate Professor

Acknowledgements

First of all, I would like to thank my advisor Dr. Brian Roark. He shared his insights and expertise with me without reservation. He provided me with lots of chances of being exposed to various very interesting problems in the field. He spent his time listening to my ideas, gave me constructive suggestions, and inspired me to materialize them. He never blamed me for the mistakes I made. He is very supportive to me all the time. Thank you very much Brian, for your endless guidance and support.

Also thanks very much to my thesis committee members for their valuable comments and suggestions. Thank you to Dr. Peter Heeman for giving me very practical and constructive suggestions about paper writing, and to Dr. Aaron Cohen for his support.

I would also like to thank my colleagues, including Dr. Kristy Hollingshead for her encouragement and intuitive discussions, and Dr. Christian Monson for the helpful conversations we had about various problems. Thanks very much to my officemates Ethan Selfridge and Rebecca Lunsford.

Many special thanks to the staff members: particularly to Pat Dickerson for administrative matters, and to Sean Farrell, who is extremely supportive. The experiments would not have been carried out smoothly if it was not Sean's kind understanding.

Moreover, thanks very much to Pattara Kiatisevi for the suggestion of Thai resources.

Finally, I would like to thank my family for their love and care. I especially want to thank my mother for her encouragement, and for her unconditional love, scarifies, and understanding; my aunt and uncle for their endless care; my cousin for her insightful comments and warm help.

Contents

Acknowledgements . . . . . **iv**

Abstract . . . . . **ix**

1 Introduction **1**

2 Background **3**

2.1 Chinese Word Segmentation 3

2.2 Related Work 4

2.3 Log-linear Model 4

2.4 Perceptron Algorithm 5

2.5 Evaluation 6

3 The System **8**

3.1 Segmentation Corpus 8

3.2 Baseline System Feature 8

3.3 Baseline System Performance 10

4 Feature Engineering **11**

4.1 Features for External Resources 11

4.2 Word Dictionary Feature 12

4.2.1 Resource Incorporation Method 12

4.2.2 Word Dictionary Feature 13

4.3 Word Forming Feature 13

4.3.1 String Repetition Pattern Feature 13

4.3.2 Affix Feature 14

4.3.3 Suoxie Feature 14

4.3.4 Feature Using Part-of-Speech (POS) Information 14

4.4 NE Feature 15

4.4.1 Japanese Name Feature 15

4.4.2 Chinese Name Feature 15

4.4.3 Transliteration Name Feature 15
4.4.4 Famous People Name Feature 16
4.4.5 Chengyu Idiom Feature 16
4.4.6 Geographical Location Feature 16
4.4.7 Other Name Entity Feature 17
4.5 Normalization 17
4.6 Features with Web Data 18
4.7 Feature Engineering Summary 18

5 System Gain Prediction 21

6 Thai Word Segmentation 25
6.1 Thai Data 25
6.2 Experiments and Results 25

7 Conclusion and Future Work 28

Biographical Note 33

List of Tables

2.1 An example of bi-tag and four-tag tag set. 4

3.1 Second SIGHAN competition corpora [3]. 8
3.2 Type I feature templates in baseline system. 9
3.3 Type II feature templates in baseline system. 10
3.4 Baseline system performance and state of the art results on PKU corpora. . 10

4.1 Results for dictionary incorporation. 19
4.2 Feature ablation results on PKU test set and the state of the art results for the open task. 20

6.1 Results of Thai word segmentation with external dictionaries. 27

List of Figures

2.1 A variant of perceptron algorithm [11]. 6

4.1 Example of independently incorporating one list. 12
4.2 Example of jointly incorporating two lists. 12

5.1 External resources' contribution prediction metric I. 22
5.2 External resources' contribution prediction metric II. 23

6.1 Histogram of Thai training corpora. 26
6.2 External resources' contribution prediction metric II on Thai. 27

Abstract

A controlled study of the contribution of external resources on Chinese word segmentation

Yongshun Chen

Supervising Professor: Brian Roark

In Chinese, written sentences consist of a concatenation of characters and punctuation with no additional word boundary delimiters. As an important first step in many Chinese natural language processing (NLP) applications, Chinese word segmentation (CWS) inserts word boundaries given unsegmented Chinese sentences. In recent international competitions on CWS, there are two modes of training and evaluation, namely *closed* and *open*. In closed task, no data nor information [3] in addition to training corpora can be used for training. In open task, any external material can be used. To our knowledge, for the open task, there has been no serious study of which external resources are useful and which are not; nor has there been any study that quantifies each resource's contribution. Moreover, given a potentially helpful resource, how it is incorporated into the system can also make a difference. We explore different resource incorporation methods to find the more helpful method. We quantify the influence of different external resources for open task, and further try to predict in advance which resource will improve performance. Empirical results show that dictionaries that are independent from the training corpora are extremely helpful to system performance. This finding is successfully generalized for the word segmentation problem of language other than Chinese. We also find that number, ASCII character, and punctuation normalization brings in additional gains.

Chapter 1

Introduction

In NLP, hand-crafted systems are no longer popular, and most reported systems nowadays depend on statistical methods. However, it is important to notice that many statistical systems still contain rich hand-crafted knowledge (which can be represented by language dependent rules, lexicons, etc.). However, there is little general understanding about the usability among different knowledge resources.

For CWS, SIGHAN (ACL's special interest group on Chinese language processing) sponsored five international competitions. The competitions allow two modes of training and evaluation, namely *closed* and *open*. In the closed task, candidates can only use the given training data for system training. In the open task, candidates can use arbitrary external resources, which must then be reported. In general, for the same training and testing data, people receive much higher accuracy in the open task than in the closed task. This argues that additional linguistic knowledge contributes to system performance. The used external resources in literature are dictionaries, family name list, etc.

To our knowledge, there is no systematic study on which resources are useful to CWS and which are not, nor the extent to which each resource contributes. Furthermore, how the resource is integrated into the system can make a difference. We do a controlled study on external resources' effect, and quantify each resource's contribution. In the literature, many systems [1, 8, 14, 17, 19, 20] have multiple stages of independent pre- or post-processing for each special word category. Different from them, we incorporate these resources as features in our training method. This allows for incorporation of arbitrary linguistic knowledge in a uniform and flexible way, eschewing the need for ad hoc pre- or post-processing that makes use of such resources. Furthermore, we try to predict in advance which resource will help.

Finally, we generalize the findings in CWS to the Thai language, which also has the issue of word segmentation, and compare improvements in system performance.

As a preview of our results, the biggest improvement to our system (1.5% F score) was obtained with an external dictionary that is independent from the training corpora, but both data follow the same segmentation standard. Furthermore, normalization of ASCII characters, punctuation, and numbers brings in additional gain (0.8% F score). Other resources fail to show significant system performance improvement independently, but when adding all these resources into the system simultaneously, we obtain another moderate improvement (0.3% F score).

It is not surprising that many researchers working on this problem have tried some of these resources in different ways with no success, but they followed the tendency in this field to not report them. This results in multiple people trying independent but similar ideas only to repeat negative results. Therefore, another contribution of this work is to study and also report exactly the maximized effectiveness of each piece of supplemental information.

Chapter 2

Background

2.1 Chinese Word Segmentation

The Chinese writing system is different from that of English. In written English, word boundaries are indicated by whitespace and punctuation. In Chinese, written sentences consist of a concatenation of characters and punctuation with no additional word boundary delimiters. For example, if we write English in the way we do for Chinese, the English sentence “He was here.” will become “Hewashere.” Therefore, the problem of CWS is to insert word boundaries given unsegmented raw Chinese sentences.

CWS is the first step of any Chinese NLP application that needs tokenized words as input. Because errors in CWS will be passed along to subsequent NLP tasks, the performance of CWS is of great importance.

We treat CWS as a tagging problem [18]. In this study, we experimented both “BI” bi-tag tag set, and “SBME” four-tag tag set. For bi-tag, for each character in the unsegmented corpora, if it is the first character of a word, it is tagged with tag “B”, otherwise tag “I”. For “SBME” four-tag, for each character, if the character itself is a single word, it is tagged “S”. If it begins a word of two or more characters, it is tagged “B”. If it is in the middle a word, namely it has at least one character to its left and one to its right, it is tagged “M”. If it ends a word of two or more characters, it is tagged “E”. An example is in Table 2.1.

Table 2.1: An example of bi-tag and four-tag tag set.

Chinese sentence	母	亲	吃	胡	萝	卜	。
segmentation	母 亲		吃	胡 萝 卜			。
bi-tag tag	B	I	B	B	I	I	B
four-tag tag	B	E	S	B	M	E	S
translation	mother		eats	carrots			.

2.2 Related Work

As we mentioned above, in recent international competitions on CWS, there are two modes of training and evaluation, namely *closed* and *open*. In a closed task, no material other than training corpora may be used for training. In open task, any information or external material [3] can be used. By its nature, the closed task encourages participants to focus on tuning machine learning methods. Discriminative methods have generally shown the best performance, and include maximum entropy [6, 10], perceptron [8, 21], and conditional random fields (CRF) [15, 17, 20, 22]. Features include word-based features [21], sub-word-based features [17, 20], and character-based features [8, 10, 15]. Further improvements have been gained by dictionary-based N-gram language models [1, 4, 7, 17, 20], and rule based dictionary matching [21]. Many systems [1, 8, 14, 17, 19, 20] benefit from independent stages of pre-processing, and post-processing. For the open task, external dictionaries [1, 6, 10, 19] have been used, as well as data segmented with different segmentation standards [1, 7, 10]. Family name lists [5, 7], transliterated name lists [7], and raw text [9] have also been used.

2.3 Log-linear Model

Following [12, 13], given a character-tag sequence $\mathbf{CT} = \mathbf{c_1}\tau_\mathbf{1}, \mathbf{c_2}\tau_\mathbf{2}, ..., \mathbf{c_k}\tau_\mathbf{k}$ as training example, where the character sequence is from the unsegmented raw corpora, and the tag sequence is the gold segmentation answer for this character sequence, an exponential model has the form

$$P(\tau|c) = \frac{\exp(\Phi(\mathbf{CT}) \cdot \overline{\alpha})}{Z(\mathbf{C})} = \frac{\exp(\sum_{i=1}^{n} \phi_i(\mathbf{c}\tau)\alpha_i)}{Z(\mathbf{C})}, \tag{2.1}$$

where $\Phi(\mathbf{CT}) \in \Re^{\mathbf{d}}$ is an n-dimensional feature vector; $\overline{\alpha} \in \Re^{d}$ is an n-dimensional weight vector; ϕ_i is the fired feature given this character sequence, and Z is a normalization factor:

$$Z(\mathbf{C}) = \sum_{\tau \in T^{|\mathbf{c}|}} \exp(\sum_{i=1}^{n} \phi_i(\mathbf{c}\tau)\alpha_i). \tag{2.2}$$

By taking the log of the probabilities in (2.1), and discarding normalization constant, we have

$$\log P(\tau|\mathbf{c}) = \Phi(\mathbf{CT}) \cdot \overline{\alpha}, \tag{2.3}$$

which is a linear combination of weighted features.

Let **GEN** be a function that enumerates tag sequences for **C**. Then the learning task is to use training corpora to set the value of weight vector $\overline{\alpha}$, so that a mapping from input **C** to output $F(\mathbf{C})$ can be found:

$$F(\mathbf{C}) = \underset{\mathbf{T} \in \mathbf{GEN}(\mathbf{C})}{\text{argmax}} \; \mathbf{\Phi}(\mathbf{CT}) \cdot \overline{\alpha}. \tag{2.4}$$

2.4 Perceptron Algorithm

Under the log-linear framework, there are various parameter estimation methods, such as conditional random fields, perceptron, maximum entropy. In this study, we adopted perceptron algorithm for training. It is an online algorithm that converges quickly during a few passes over the training corpora.

We follow a variant of the perceptron algorithm [11] given in Figure 2.1. In the training phase, for each unsegmented input character sequence, system temporarily outputs a tag sequence based on the model's features' weights at that time. Then for each difference between the temporary output and the correct answer of this character sequence, each feature associated with the temporary output will be fired, and have its weight subtracted

```
**Inputs:** Training examples $(x_i, \tau_i)$
**Initialization:** Set $\overline{\alpha} = 0$
**Algorithm:**
For $t = 1 \ldots T, i = 1 \ldots N$
Calculate $z_i = \text{argmax}_{z \in GEN(x_i)} \Phi(x_i, z) \cdot \overline{\alpha}$
If$(z_i \neq \tau_i)$ then $\overline{\alpha} = \overline{\alpha} + \Phi(x_i, \tau_i) - \Phi(x_i, z_i)$
**Output:** Parameters $\overline{\alpha}$
```

Figure 2.1: A variant of perceptron algorithm [11].

by one. Similarly, each feature associated with the correct answer will be fired, with its feature weight added by one. After each iteration over the training corpora, we test on a development set and track system performance. The model converges when accuracy fails to increase during a few (in our experiments, we choose five) consecutive iterations. We use Viterbi decoding in the training phase, and forward-backward decoding in the testing phase.

One problem of this training method is that the trained model is prone to be over-trained on the training corpora. In other words, system performance can be promising when evaluated on development set, but it tends to drop more than expected when evaluated on testing set. To resolve this, after the model converged, the set of averaged (for each feature independently) cumulative (over iterations) feature weights [2] is adopted as feature weights in the final model.

2.5 Evaluation

We use F score to report system performance. For some experiments, we additionally report out-of-vocabulary (OOV) word recall, in vocabulary (IV) word recall, or use cross validation.

F Measure: F-measure accuracy is the weighted harmonic mean of precision and recall:

$$F = 2 \cdot \frac{precision \cdot recall}{precision + recall}. \tag{2.5}$$

Precision is the number of correctly identified words divided by word count in system output. Recall is the same denominator divided by word count in the correct answer of the input corpora.

OOV Words Recall: OOV words are the words that are not in the system training data, but in the data on which the system is evaluated. How well does the system handles OOV words can largely influence overall system performance. In fact, one goal of incorporating external knowledge is to enhance the system's ability of OOV words identification. Therefore, we evaluate recall on OOV words, which is the ratio between the number of correctly identified OOV words, and the total number of OOV words in the correct answer.

IV Words Recall: IV words are the words that occur in both training set and the evaluation data set. Similar to OOV words recall, we define IV words recall as the ratio between the number of correctly identified IV words, and the total number of IV words in the correct answer.

Cross Validation: We apply ten-fold cross validation to some experiments. We partition the training set into ten subsets, and act accordingly to the corresponding correct answer. Then by using each subset as temporary development set, and the remaining as temporary training set, we separately train ten temporary converged models, which are ten sets of weighted features. Finally, we combine these ten temporary models to derive the final model: for each feature, we use arithmetic mean of its ten temporary weights as its final feature weight.

Chapter 3

The System

3.1 Segmentation Corpus

In this study, we report on the PKU CWS corpora, released by Peking University for the second SIGHAN competition (see Table 3.1). It follows GB13715[1] segmentation standard, and is widely used for CWS studies. Comparing with other corpura in Table 3.1, it has higher OOV rate and bigger test set.

Table 3.1: Second SIGHAN competition corpora [3].

corpora	**encoding**	**training size (words/types)**	**testing size (words/types)**	**OOV rate**
AS	Big Five Plus, Unicode	5.45M / 141K	122K / 19K	0.043
PKU	CP936, Unicode	1.1M / 55K	104K / 13K	0.058
CityU	Big Five, Unicode	1.46M / 69K	41K / 9K	0.074
MSR	CP936, Unicode	2.37M / 88K	107K / 13K	0.026

3.2 Baseline System Feature

In our study, we define *class* based on tags. Two classes are involved: class "B" and class "I". For "BI" tag set, tag "B" is mapped to class "B", and tag "I" is mapped to class "I". For "SBME" tag set, both tag "S" and tag "B" are mapped to class "B", and both tag "M" and tag "E" are mapped to class "I".

[1]GB13715 is the national standard for text segmentation in computer applications in China.

In our perceptron baseline system, we have two types of features. Type I features only contain character and tag information. Type II features must contain tag class, and character and tag information is optional. At run time, the system automatically generates each feature (regardless of feature type) given a specific feature template and the specific input sequences.

- **Type I Feature Templates**

  Let c_0 be the current character, c_{-1} the character immediate left to c_0, c_1 the character immediate right to c_0, τ_0 the tag of c_0, and τ_{-1} the tag of c_{-1}. Baseline system's type I feature templates are in Table 3.2, where "ID" is the index for each feature template. We found this set of feature templates by using a greedy approach. We started with a small set of feature templates (feature template 1 to 10) as the final feature template set, and experimenting one additional feature template a time. The additional feature template that yields the biggest system performance gain is then inserted to the final feature template set. By doing this iteratively, we obtained the final 14 feature templates.

Table 3.2: Type I feature templates in baseline system.

ID	**feature template**	**ID**	**feature template**	**ID**	**feature template**	**ID**	**feature template**
1	$\tau_0 c_0$	5	$\tau_0 c_{-1}$	9	$\tau_0 c_{-1} c_0$	13	$\tau_0 c_{-1} c_0 c_1$
2	$\tau_{-1} \tau_0 c_0$	6	$\tau_{-1} \tau_0 c_{-1}$	10	$\tau_{-1} \tau_0 c_{-1} c_0$	14	$\tau_{-1} \tau_0 c_{-1} c_0 c_1$
3	$\tau_0 c_1$	7	$\tau_0 c_0 c_1$	11	$\tau_0 c_{-2} c_{-1}$		
4	$\tau_{-1} \tau_0 c_1$	8	$\tau_{-1} \tau_0 c_0 c_1$	12	$\tau_{-1} \tau_0 c_{-2} c_{-1}$		

- **Type II Feature Templates**

  Based on the terminology of type I feature templates, let C_0 be the tag class of tag τ_0, and C_{-1} the tag class of tag τ_{-1}. Starting with the 14 feature templates of type I, similar to how we found the set of type I feature templates, we again use a greedy approach. We experimented inserting one type II feature a time. The type II feature template that yields the biggest system performance gain is then kept in the final type II feature template set. By doing this iteratively, we obtained the final five type II feature templates given in Table 3.3, where "ID" is the index for each feature template.

Table 3.3: Type II feature templates in baseline system.

ID	**feature template**	**ID**	**feature template**
1	$C_{-1}C_0c_{-1}$	4	$C_{-1}\tau_{-1}c_{-2}c_{-1}$
2	$C_{-1}C_0c_{-1}c_0$	5	$C_{-1}C_0\tau_{-1}c_{-2}c_{-1}$
3	$C_{-1}C_0\tau_0c_{-1}c_0$		

3.3 Baseline System Performance

We split the prescribed PKU corpora into two portions: the first portion is the first 17149 lines, and we use it as training set to train the system; the remaining 1907 lines are held out as development set, and we use it for system convergence decision. In our training set, there are 4698 character types, and 1826448 characters. Our test set is the given test set in the PKU corpora. Baseline system performance and state of the art [14, 10] performance are in Table 3.4.

We tune the feature template sets with four-tag tag set, and it achieved higher baseline performance than the bi-tag tag set. Therefore, in subsequent experiments, we report results with four-tag tag set unless otherwise explained.

Table 3.4: Baseline system performance and state of the art results on PKU corpora.

evaluation data	**type**	**F %**	**P %**	**R %**	R_{IV}	R_{OOV}
DEV	closed, bi-tag	95.9	95.7	96.0	96.1	93.9
DEV	closed, four-tag	96.1	95.9	96.3	96.3	94.1
test	closed, bi-tag	93.1	92.1	94.0	94.3	56.5
test	closed, four-tag	**94.0**	93.2	94.7	94.8	67.0
test[14]	closed, state of the art	95.2	95.6	94.8	not reported	77.8
test[10]	open, state of the art	96.9	96.9	96.8	97.6	83.8

Chapter 4

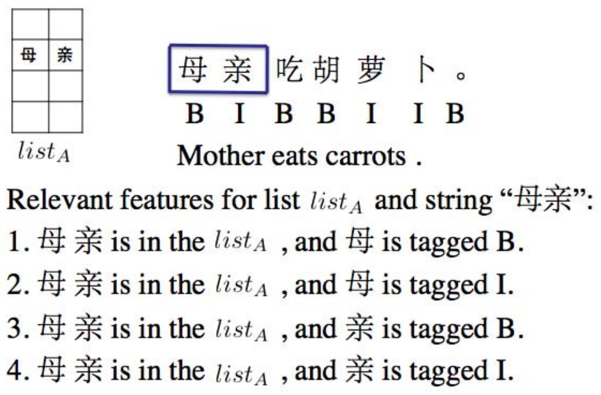
Feature Engineering

4.1 Features for External Resources

We represent each external resource as lists of finite number of character strings, and then incorporate these lists into system by features. Each feature contains character and tag information.

To incorporate a single list of string, where each string has N characters, we encode $2N$ features for the case of bi-tag tag set. For example, suppose N is two, for any specific input string (following the denotation in chapter 3.2, the input string is denoted as a sequence of c_i, $i = -1, 0, 1, ...$), we have four features: c_0c_1 is in the list with c_0 tagged "B" in the input string, c_0c_1 is in the list with c_0 tagged "I" in the input string, $c_{-1}c_0$ is in the list with c_0 tagged "B" in the input string, $c_{-1}c_0$ is in the list with c_0 tagged "I" in the input string. An example is given in Figure 4.1. For the case of four-tag tag set, we similarly use $4N$ features. We use this method for resources such as dictionary, name entity (NE), etc.

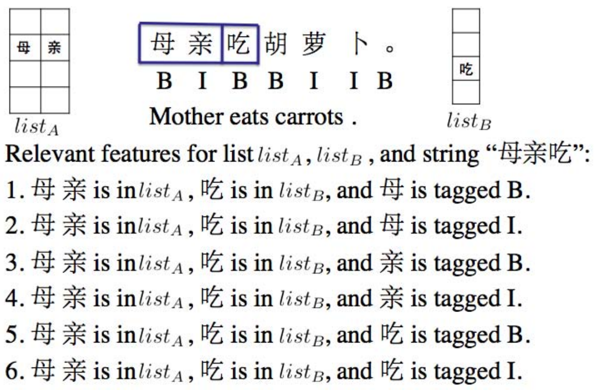
To jointly incorporate two lists, denoted "list_A", which is a list of string, where each string has M characters, and "list_B", which is a list of string, where each string has N characters, for the bi-tag case, we encode $2M + 2N$ features. For example, if M is one and N is two, for any specific input string (follow the denotation in chapter 3.2), we have six features: c_0 in "list_A" and c_1c_2 in "list_B" with c_0 tagged "B" in the input string, c_0 in "list_A" and c_1c_2 in "list_B" with c_0 tagged "I" in the input string, c_{-1} in "list_A" and c_0c_1 in "list_B" with c_0 tagged "B" in the input string, c_{-1} in "list_A" and c_0c_1 in "list_B" with c_0 tagged "I" in the input string, c_{-2} in "list_A" and $c_{-1}c_0$ in "list_B" with c_0 tagged



Relevant features for list $list_A$ and string “母亲”:
1. 母 亲 is in the $list_A$, and 母 is tagged B.
2. 母 亲 is in the $list_A$, and 母 is tagged I.
3. 母 亲 is in the $list_A$, and 亲 is tagged B.
4. 母 亲 is in the $list_A$, and 亲 is tagged I.

Figure 4.1: Example of independently incorporating one list.

“B” in the input string, c_{-2} in “list_A” and $c_{-1}c_0$ in “list_B” with c_0 tagged “I” in the input string. An example is given in Figure 4.2. For the case of four-tag, we similarly use $4M + 4N$ features. We use this method to encode affix, and people’s full name, etc.



Relevant features for list $list_A$, $list_B$, and string “母亲吃”:
1. 母 亲 is in$list_A$, 吃 is in $list_B$, and 母 is tagged B.
2. 母 亲 is in$list_A$, 吃 is in $list_B$, and 母 is tagged I.
3. 母 亲 is in$list_A$, 吃 is in $list_B$, and 亲 is tagged B.
4. 母 亲 is in$list_A$, 吃 is in $list_B$, and 亲 is tagged I.
5. 母 亲 is in$list_A$, 吃 is in $list_B$, and 吃 is tagged B.
6. 母 亲 is in$list_A$, 吃 is in $list_B$, and 吃 is tagged I.

Figure 4.2: Example of jointly incorporating two lists.

In the subsequent part of this thesis, we follow these feature encoding methods unless otherwise additionally explained.

4.2 Word Dictionary Feature

4.2.1 Resource Incorporation Method

We tried different ways of dictionary incorporation, to find the external resource incorporation method that can better reveal a given resource’s contribution. One method is to use the method given in chapter 4.1. Another method is to append the dictionary to the end of the training data. Using the same dictionary, we found that the method in chapter 4.1 outperforms the other method. Preliminary results are in chapter 4.7.

4.2.2 Word Dictionary Feature

We experiment several dictionaries: a dictionary derived from the training corpora, a dictionary derived from the MSR corpora of the same competition, a dictionary derived from Penn Chinese Treebank (PCTB), and a dictionary that follows the same segmentation standard as the training corpora, but is independent from them.

Given multiple helpful word dictionaries that follow different segmentation standards, we try to combine their contributions together. One way is to have separate features for each dictionary, and include all these features in the system in parallel. Another way is to concatenate these dictionaries together, and encode features for the obtained big dictionary.

Preliminary results are in chapter 4.7.

4.3 Word Forming Feature

4.3.1 String Repetition Pattern Feature

String (normally contains one or two characters) repetitions commonly exist in written Chinese. The string repetitions obey several patterns, which are consistently segmented[1]. Take pattern ABAB for example: if "A" and "B" represents different characters, string "ABAB" is always segmented into two "AB" words. For example, "高兴高兴" (meaning: to have joy) is the repetition of "高兴" (meaning: happy), and it is segmented into "高兴 高兴" (suppose empty space is the word boundary delimiter). Similar segmentation consistencies also exist in pattern AAB, AABB, ABA, ABAB, ABAC, ABB. We encode features for each pattern, and features for the optional suffixes (的 and 地, which hardcodes the pattern's Part-of-Speech as adjective and adverb, respectively) of pattern AA and AABB.

[1]Please refer to the segmentation guideline (section 3.1) of the prescribed PKU corpora.

4.3.2 Affix Feature[2]

Affixes in Chinese extend and assist word usage in dialects, syntactical variations, and ways of referring to people with especial emphasis (such as respect, warmth, order in the family, etc.). For example, the Chinese may call his or her aunt (in Chinese: 姨) who is the second (in Chinese: 二) daughter of her family 二姨. We encode features to incorporate lists of common prefix (老, 大, 小, 阿, 超, 非, etc.), suffix (们, 家, 学, etc.), and infix (了一) into the system.

4.3.3 Suoxie Feature

Suoxie is a special form of abbreviation – key characters extracted from word(s) to replace the word(s) with unchanged meaning. For example, the Chinese can use 京津唐 to represent the geographical area that covers 北京, 天津, and 唐山. As our NE resources already include many suoxie words, we only encode features for souxie of province and city name list. With a list of each province's, and each common city's abbreviation, we encode it jointly with itself to represent neighboring abbreviation concatenation.

4.3.4 Feature Using Part-of-Speech (POS) Information

In Chinese, compound word contains two or more morphemes, whose POS is up to word compounding rules[3]. For example, word 苦瓜 (meaning: balsam pear, bitter gourd) obeys the rule "one-character adjective followed by one-character noun", where 苦 means bitter, and 瓜 means melon. With POS tagged data[4] that follows the same segmentation standard as the PKU corpora, we collect separate list for each morpheme class. Then we jointly incorporate two lists with respect to word compounding rules.

[2] This is according to the segmentation guideline (chapter 3.2) of the prescribed PKU corpora.
[3] This is according to the segmentation guideline (chapter 3.3) of the prescribed PKU corpora.
[4] http://download.csdn.net/source/1259040

4.4 NE Feature

4.4.1 Japanese Name Feature

In Chinese language, Japanese name is written as given name following family name (except the emperor). We collect a list of 8.7K[5] Japanese family name, and a list of common Japanese female given name[6]. Then we incorporate them separately as a single list, as well as jointly to represent full name.

4.4.2 Chinese Name Feature

Chinese name has the form of family name (over 85% has one character) followed by given name (mostly has one or two characters). We have a list of 373 common Chinese family names (97% has one character), and a list of 6.5K common Chinese given names (of both one and two characters).

In addition to having all the features we have for Japanese names for Chinese names, we further have features for Chinese full name before a punctuation, features for full name list, and features for a common custom – the first character of a given name is also a family name, etc.

4.4.3 Transliteration Name Feature

Chinese generally transliterates foreign names by picking the sequence of characters whose pronunciation in Chinese is phonetically similar to the name's pronunciation in the source language. Thus “巴巴拉 or 芭芭拉” (both are common Chinese translations of name Barbara, and pronounced ba-ba-la in Chinese) approximates the pronunciation of “Barbara”. Certain Chinese characters are frequently used for this purpose than most others.

With 66K Chinese translation of foreign names, we extract a list of the first one character, and a list of first two characters sequence, same for the last characters, and another list of all used characters. Then we include these lists jointly (such as first two

[5]http://zh.wikipedia.org

[6]http://hi.baidu.com/%D3%EA%BD%A5/blog/item/4688aab4c2349f748ad4b2b0.html

character sequence list with the character list) to generalize for unknown transliterated names.

4.4.4 Famous People Name Feature

As external resource, we collect[7] lists of names of world famous philosophers, singers, musicians, scientists, ideologists, writers and their pen names, and names of Chinese political leaders since year 1954. We also have a list for the "dot" symbol, which is sometimes observed in the middle of a name. Then we group[8] these names into lists with respect to segmentation convention difference. We include each list separately, and we also include name list and the dot symbol list jointly for the purpose of generalization.

Furthermore, we jointly incorporate Chinese family name list and person's title list to encode a way of referring to person – family name followed by title. For example, professor (in Chinese: 教授) Wang (in Chinese: 王, which is the family name of the faculty) is written as 王教授 in Chinese.

4.4.5 Chengyu Idiom Feature

Chengyu (such as 心想事成, which means all dreams come true) is a kind of conventional Chinese idiom. 96% of chengyu has four characters. We incorporated a list of 30K[9] chengyu idiom into our system.

4.4.6 Geographical Location Feature

We have lists of world countries, cities, mountains, basins, islands, peninsulas, rivers, lakes, seas, oceans, continents, and Chinese provinces, springs, dams.

[7]All resource for chapter 4.5.4 to chapter 4.5.7 are download from http://www.wikipedia.org.

[8]Please refer to the segmentation guideline (chapter 2.2.1) of the prescribed PKU corpora.

[9]http://www.cnpoem.net/cy

4.4.7 Other Name Entity Feature

At international level, we gather lists of companies names, languages, awards, religions, music types, musical instruments, currencies, news agencies, and measurement units. Limited to China, we collect lists of operas, 56 nationalities, highway lines, train lines, festivals, democratic parties and their "suoxie", colleges' names, all government departments' names, and lists of Chinese year denotation, namely "ganzhi (such as 甲子)" (denote year with 60 year cycle), "shengxiao (such as 亥猪)" (denote year with 12 year cycle), and Chinese dynasty names. Furthermore, we have lists of vegetables, fruits, vitamins, and weather conditions names. Each list is included into the system independently.

4.5 Normalization

Different from previously mentioned cases, segmentation decisions for ASCII strings, numbers[10], and punctuation are more dependent on their neighboring context, rather than the exact symbol or character. Therefore, we normalize for ASCII strings, numbers, and punctuation.

For ASCII strings, we have features for ASCII and Chinese character boundaries (c_{-1} is a Chinese character and c_0 is an ASCII character; c_{-1} is an ASCII character and c_0 is a Chinese character), ASCII character and number boundaries (c_{-1} is an ASCII character and c_0 is a number symbol; c_{-1} is a number symbol and c_0 is an ASCII character), ASCII character and punctuation boundaries (c_{-1} is an ASCII character and c_0 is a punctuation symbol; c_{-1} is a punctuation symbol and c_0 is an ASCII character), as well as for ASCII character concatenation (both c_{-1} and c_0 are ASCII characters). Features are encoded by joint lists.

For punctuation, we follow the feature encodings for ASCII strings, except that features for punctuation and ASCII character boundaries are removed, as they already exist as features for ASCII strings.

For numbers, we create a list of prefix (denoted prefix_list_1) and a list of suffix near any of witch number needs to be segmented apart, and another list of prefix and another

[10]Please refer to the segmentation guideline (chapter 2.2.5, 2.2.6) of the prescribed PKU corpora.

list of suffix near any of which number needs not with respect to the segmentation guideline. Then we encode corresponding features for each situation (such as $c_{-2}c_{-1}$ are in prefix_list_1, and c_0 is in the number symbol list).

4.6 Features with Web Data

All previous resources are sets of lexicons that we collect for specific topics or categories. In this section, we make an effort to use raw web data, and search term[11], to try to automatically extract word boundary information.

With 1.1G news from different websites between 2008 May and June, we separately extract lists of one character before punctuation, two character sequence before punctuation, one character after punctuation, two character sequence after punctuation, and the concatenation of the character that is immediately preceding and following each punctuation.

From Sogou search engine's half years' search term, we separately extract list of search terms of two, and of three characters.

4.7 Feature Engineering Summary

Preliminary results show that all the *external* dictionaries, namely the dictionaries derived from data other than the training corpora, do not hurt system performance, and most of them make contributions (only the dictionary derived from PCTB failed to make significant improvement). The contribution turns out to be the biggest when the dictionary and training corpora follow the same segmentation standard.

Given multiple helpful dictionaries that follow different segmentation standards, we try to combine them by concatenating all helpful dictionaries into a new dictionary, which then be incorporated into the system. We also try using all these dictionaries in parallel. However, it is hard to say whether dictionary combination can outperform the most helpful dictionary (in the set of helpful dictionaries) or not.

[11]Both web data and search term are downloaded from http://www.sogou.com/labs.

We also find that the word dictionary derived from the training corpora hurts system performance, even with complicated cross validation scenarios. It is because in this case, features related to this dictionary are excessively reliable at training time. Hence, they are the only features that are well trained, leaving the massive amount of baseline features fail to be well tuned. At testing time, however, this dictionary no longer has its perfect word coverage, so the performance drops.

Results for dictionary incorporation are given in Table 4.1. It also shows that the method we adopt to incorporate external dictionary yields better system performance (comparing with appending the dictionary to the end of training corpora). Note that dictionary "D_1" is derived from the training corpora. Dictionary"D_2" is from PKU[12] but independent from training corpora. Dictionary "D_3" is from the MSR corpora released by second SIGHAN competition. Dictionary "D_4" is from PCTB. Dictionary "D_5" is the concatenation of dictionary D_2, D_3, and D_4. "D_2 (append)" indicates incorporating dictionary "D_2" by appending it to the end of training corpora.

Table 4.1: Results for dictionary incorporation.

condition	**four-tag tag set**					**bi-tag tag set**				
	F%	**R%**	**P%**	R_{IV}	R_{OOV}	**F%**	**R%**	**P%**	R_{IV}	R_{OOV}
baseline	94.0	93.2	94.7	94.8	67.0	93.1	92.1	94.0	94.3	56.5
baseline + D_1	92.3	93.3	91.2	97.1	31.5	91.5	92.5	90.5	97.8	22.7
baseline + D_2 (append)	93.6	92.8	94.5	94.4	65.8	92.8	91.7	94.0	94.0	54.2
baseline + D_2	**95.5**	95.3	95.7	97.0	68.7	**94.4**	94.2	94.7	96.4	58.8
baseline + D_3	94.7	94.2	95.2	95.7	69.0	93.8	93.2	94.3	95.4	58.4
baseline + D_4	94.1	93.7	94.4	95.5	64.1	93.9	93.4	94.3	95.2	62.8
baseline + D_2 + D_3 + D_4	94.6	94.9	94.3	97.0	60.0	94.6	94.6	94.5	96.7	60.5
baseline + D_5	**94.9**	95.0	94.7	96.8	65.3	**94.9**	94.9	94.8	96.6	67.4

[12]http://ccl.pku.edu.cn/doubtfire/Course/Chinese%20Information%20Processing/2002_2003_1.htm

With all the external resources, our system achieved 96.6% F score, which is very close to the state of the art [10] performance for the open task. Preliminary results show that the external dictionary that is not derived from the training corpora, but follows the same segmentation standard as it made the biggest contribution. All normalization features made an additional contribution. All other resources fail to make independent contribution, but they together yielded another moderate improvement. A summary of feature ablation results is in Table 4.2, where "norm" indicates all normalization features, and "others" indicates features other than dictionary and normalization features.

List boundary features (features focusing on the tag of the character that is immediately proceeding or following a list token) did not make any statistical significant improvement. Ten-fold cross validation as another evaluation method showed no improvement on system performance.

Table 4.2: Feature ablation results on PKU test set and the state of the art results for the open task.

experiment condition	**F%**	**R%**	**P%**	R_{IV}	R_{OOV}
baseline	94.0	93.2	94.7	94.8	67.0
baseline + D_2	95.5	95.3	95.7	97.0	68.7
baseline + norm	94.6	94.0	95.3	95.1	75.1
baseline + D_2 + norm	96.3	96.2	96.4	97.4	76.5
baseline + D_2 + norm + others	**96.6**	96.5	96.6	97.5	79.6
baseline + D_2 + others	95.6	95.6	95.7	97.1	70.2
baseline + norm + others	95.3	94.7	95.8	95.8	77.1
baseline + others	94.5	93.9	95.2	95.4	69.9
state of the art for open task [10]	96.9	96.9	96.8	97.6	83.8

Chapter 5

System Gain Prediction

Given a set of external resources, it is very useful if we can predict how well each resource can help system performance in relevant to other resources. Therefore, we try to define some metrics for the resource usage prediction, with 26 lists (all external dictionaries, Japanese and Chinese family name and given name, idiom, people, pattern, geographical location, music, festival lists, etc.) among the lists we use in this study. We assume that the unsegmented test set is given. Some of our features (such as normalization features) only capture the segmentation decision of a sub-word (such as left or right boundaries). In this case, a correctly identified word, which is the smallest unit to F measure, is the result of multiple features' and lists' simultaneous contribution. We did not include such complicated cases.

We define three cases for strings, given the specific training set, raw test set, and an external resource list:

case 1: string that is in the list, in the raw test set, but not in the segmented training set.

case 2: string that is in the list, and in the segmented training set.

case 3: string that is in the list, and in the unsegmented training set.

For each string in the list, we count the summation of its occurrence in the designated training set, then accumulate all these counts, and denote the result as *token*. We also count the number of strings that exist in the list, and also exist at least once in the designated training set, and denote this count as *type*. For each case, we calculate its *token* and *type* separately. For denotation, take the case 1 for example, we use $token(case_1)$ for the former accumulated string tokens, and $type(case_1)$ for the latter unique string types.

- **Resource Contribution Prediction Metric I**

  In this metric, we define

$$score_I = \begin{cases} \frac{type(case_1)}{token(case_1)}, & \text{if } token(case_1) \neq 0 \\ 0, & \text{otherwise} \end{cases}. \tag{5.1}$$

The denominator is the count of situations in which features related to this list fire at testing time. This score describes the stability of the list's help to the OOV words. Thus help is more stable when $score_I$ is higher (which means that for the same value of $token(case_1)$, it is associated with more unique string types), because even if the system fails to segment some OOV words correctly, it still can possibly segment other unique OOV words with success.

Figure 5.1 (in which all dictionaries are external) indicates the relationship between $score_1$ and the system performance in F score. Each data point represents a list. Hence this metric is not quite successful, as for similar $score_1$ scores (of the four dictionaries), they result in significantly different system performances.

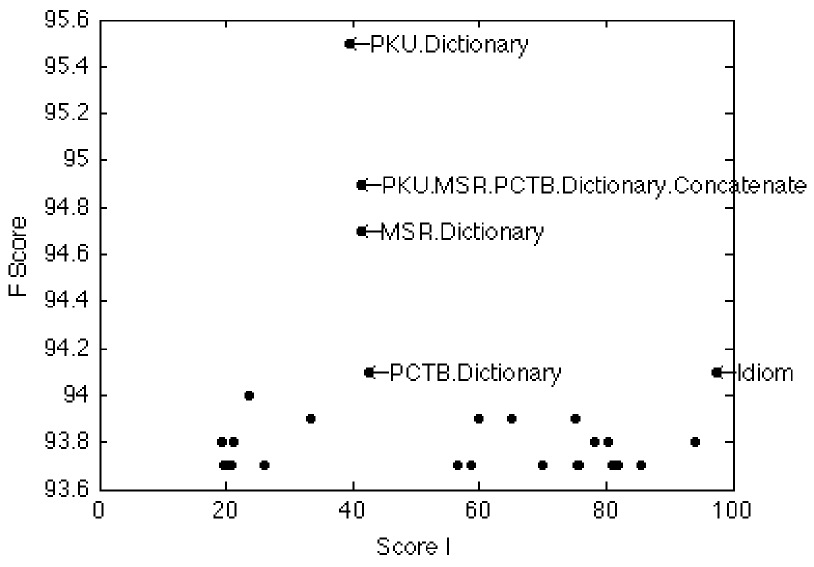


Figure 5.1: External resources' contribution prediction metric I.

Therefore, to predict a list's influence on system performance, we may also need to know the list tokens' relationship with the training corpora: such as whether features related to the list are well trained, and even if so, whether they are reliable to the system, due to the fact that the list tokens may follow different segmentation standard(s) from the training corpora.

- **Resource Contribution Prediction Metric II**

  In this metric, we define

$$score_{II} = \begin{cases} \frac{type(case_1)}{token(case_1)} \cdot \frac{token(case_2)}{token(case_3)} \cdot type(case_2), & \text{if } token(case_1) \cdot token(case_3) \neq 0 \\ 0, & \text{otherwise} \end{cases} . \tag{5.2}$$

  In this metric, beyond the information in $score_I$, we included the term $type(case_2)$, which captures the list's overlap with the words in the training corpora. This reveals how well features of this list is trained in the system. The term $\frac{token(case_2)}{token(case_3)}$ captures the extent to which different segmentation decisions apply to the same string pieces.

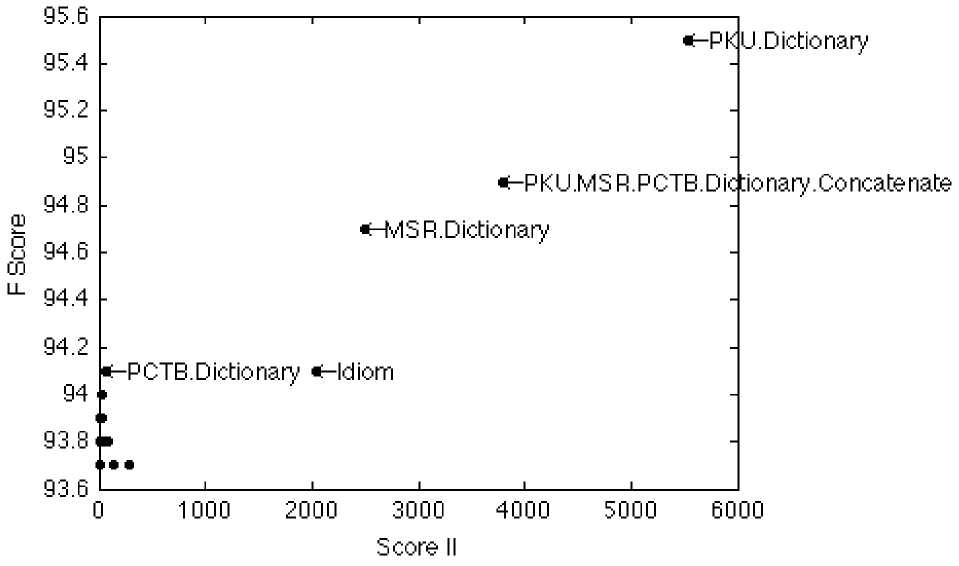


Figure 5.2: External resources' contribution prediction metric II.

From Figure 5.2 (in which all dictionaries are external) we can see that in this metric, the score increase monotonically with F score. Each data point represents a list. Hence with this metric, the resource with higher $score_2$ is prone to make bigger contribution.

Chapter 6

Thai Word Segmentation

To validate that our findings in CWS is language independent, we try to apply our findings (external dictionaries and ASCII characters, numbers, punctuation normalization are the most helpful to increase system performance) in CWS to Thai language word segmentation. Because generating resources lists for relevant normalization requires knowledge of characters, we only experiment the external dictionaries' contributions.

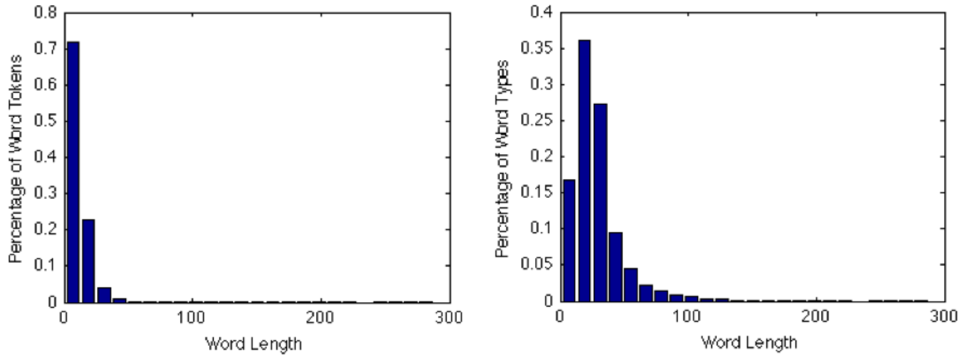
6.1 Thai Data

We use the news training set provided by the Thai word segmentation competition InterBEST[1] 2009. It has 1353967 words, 35211 word types, 5667333 characters, and 162 character types. We split the training set into two portions: the first portion is the first 208118 sentences, and we use it to train the Thai word segmenter; the remaining 26014 lines are held out as development set, and we use it for system evaluation. Histogram plots of the training corpora are in Figure 6.1.

6.2 Experiments and Results

We experimented Thai word segmentation with both "BI" bi-tag tag set, and "SBME" four-tag tag set. For each dictionary, we only include words whose lengths are smaller than 17. The baseline system has exactly the same sets of feature templates as the CWS baseline does. We obtained nine independent external dictionaries from five research

[1]http://thailang.nectec.or.th/interbest



(a) Histogram of percentage of word tokens. (b) Histogram of percentage of word types.

Figure 6.1: Histogram of Thai training corpora.

centers, which are LEXiTRON[2], longdo[3] (English-Thai, Japanese-Thai, German-Thai, French-Thai dictionaries), cettex[4](old and new dictionaries), SWATH[5], and libthai[6]. After integrating each dictionary into the baseline system in the same way as the Chinese external dictionary was, we obtain significant system gain for each dictionary with both bi-tag and four-tag tag set. Selected results on the development set are in Table 6.1 (in which all dictionaries are external), where "all" indicates all mentioned nine dictionaries, and"cttex (new)", "LEXiTRON" are the two dictionaries that made the biggest independent contribution for both tag sets.

Then we try to apply the system gain prediction metric II (presented in chapter 5) to Thai word segmentation's four-tag results (see Figure 6.2). As we see in the figure, the plot fails to be as clear and intuitive as what we obtained for Chinese. This preliminary result shows that predicting the resources' usabilities for Thai is not as straightforward as for Chinese language.

Firstly, this might be due to the differences of the length of word sequences between two languages. In PKU corpora, 97% of word tokens, and 99.3% of word types have length

[2]http://lexitron.nectec.or.th

[3]http://dict.longdo.com

[4]http://fr2.rpmfind.net/linux/rpm2html/search.php?query=cttex

[5]http://www.cs.cmu.edu/ paisarn/software.html

[6]http://linux.thai.net/projects/libthai

Table 6.1: Results of Thai word segmentation with external dictionaries.

condition	**four-tag**			**bi-tag**		
	F%	**R%**	**P%**	**F%**	**R%**	**P%**
baseline	91.8	92.3	91.3	88.6	89.5	87.6
baseline + cttex (new)	93.1	93.5	92.7	91.8	92.3	91.2
baseline + LEXiTRON	93.0	93.4	92.6	91.3	91.8	90.7
baseline + all (in parallel)	93.6	94.0	93.1	92.6	93.2	92.0
baseline + all (cat together)	92.9	93.3	92.5	91.2	91.7	90.6

no greater than five. In Thai, however, we can see from Figure 6.1 that the size of word length is generally much larger. Secondly, this may be because of the differences in the alphabet size between two languages: Thai's alphabet set (less than 200) is much smaller than that of Chinese (around 5K in PKU corpora). Finally, for the experiments we have for Thai word segmentation, each dictionary helps system performance. In this case, we are in lack of exemplars that fail to make contribution.

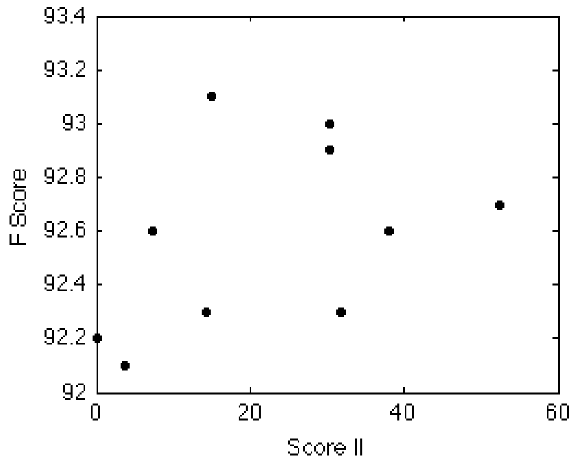


Figure 6.2: External resources' contribution prediction metric II on Thai.

Chapter 7

Conclusion and Future Work

In this work, we explored a variety of external resources' contributions to CWS. We found that dictionaries that are not derived from the training corpora are the biggest contributor to system performance, and thus finding is successfully generalized in Thai word segmentation. The contribution is the biggest when the dictionary and the training corpora follow the same segmentation standard. Dictionary that is derived from the training corpora hurts system performance, even under complicated cross validation scenarios. Also, we found that normalization of ASCII characters, punctuation, and numbers can increase system performance.

All other resources do not bring in independent system gain, but they together make a moderate additional contribution. For some resources (such as Japanese given name), the test set does not have sufficient relevant OOV words to move the system performance significantly. For some other resources (such as Chinese family name and Chinese given name), the words in the list occur massively frequent in the training corpora, so that the relevant features become unreliable through training.

Given an unknown resource, we tried to predict its usefulness in advance. It turned out that our prediction metric for CWS failed to generalize well for Thai word segmentation. The reason is unclear to us. It might be because of the difference of word length or alphabet size between the two languages. Also, for Thai, as all dictionaries help the system performance, we do not have exemplars that do not help. All these make the resource usage prediction cross two languages more difficult.

In the future, we can further work on understanding the key differences between the word segmentation tasks of Chinese and Thai. This will enable us to better characterize which resources help and which do not, and further their potential contributions.

To improve the reliability of resources, we can further apply word clustering [16] to resources prior to incorporating them into the system. Moreover, under the existing flexible framework, we can further build a joint model for CWS and Chinese POS tagging.

Bibliography

[1] Chen, A., Zhou, Y., Zhang, A., and Sun, G. Unigram language model for chinese word segmentation. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing* (2005), pp. 138–141.

[2] Collins, M. Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. In *EMNLP '02: Proceedings of the ACL-02 conference on Empirical methods in natural language processing* (Morristown, NJ, USA, 2002), Association for Computational Linguistics, pp. 1–8.

[3] Emerson, T. The second international chinese word segmentation bakeoff. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing* (2005), pp. 123–133.

[4] Fu, G., Luke, K.-K., and WONG, P. P.-W. Description of the hku chinese word segmentation system for sighan bakeoff 2005. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing* (2005), pp. 165–167.

[5] Gao, J., Li, M., Wu, A., and Huang, C.-N. Chinese word segmentation and named entity recognition: A pragmatic approach. *Comput. Linguist. 31*, 4 (2005), 531–574.

[6] Jiang, W., Zhao, J., Guan, Y., and Xu, Z. Chinese word segmentation based on mixing model. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing* (2005), pp. 180–182.

[7] Li, H., Dong, Y., Mao, X., Wang, H., and Liu, W. Chinese word segmentation in ftrd beijing. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing* (2005), pp. 150–153.

[8] Li, Y., Miao, C., Bontcheva, K., and Cunningham, H. Perceptron learning for chinese word segmentation. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing* (2005), pp. 154–157.

[9] Li, Z., and Sun, M. Punctuation as implicit annotations for chinese word segmentation. *Comput. Linguist. 35*, 4 (2009), 505–512.

[10] Low, J. K., Ng, H. T., and Guo, W. A maximum entropy approach to chinese word segmentation. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing* (2005), pp. 161–164.

[11] Roark, B., Saraclar, M., and Collins, M. Corrective language modeling for large vocabulary asr with the perceptron algorithm. In *PROC. ICASSP* (2004), pp. 749–752.

[12] Roark, B., Saraclar, M., Collins, M., and Johnson, M. Discriminative language modeling with conditional random fields and the perceptron algorithm. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics* (Morristown, NJ, USA, 2004), Association for Computational Linguistics, p. 47.

[13] Roark, B., and Sproat, R. W. *Computational Approaches to Morphology and Syntax*. Oxford University Press, New York, 2007.

[14] Sun, X., Zhang, Y., Matsuzaki, T., Tsuruoka, Y., and Tsujii, J. A discriminative latent variable chinese segmenter with hybrid word/character information. In *NAACL '09: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (Morristown, NJ, USA, 2009), Association for Computational Linguistics, pp. 56–64.

[15] Tseng, H., Chang, P., Andrew, G., Jurafsky, D., and Manning, C. A conditional random field word segmenter for sighan bakeoff 2005. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing* (2005), pp. 168–171.

[16] Turian, J., Ratinov, L., and Bengio, Y. Word representations: a simple and general method for semi-supervised learning. In *ACL '10: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics* (Morristown, NJ, USA, 2010), Association for Computational Linguistics, pp. 384–394.

[17] Wang, Z., Huang, C., and Zhu, J. Which performs better on in-vocabulary word segmentation: Based on word or character? In *Proceedings of the Sixth SIGHAN Workshop on Chinese Language Processing* (2008), pp. 61–68.

[18] Xue, N., and Shen, L. Chinese word segmentation as lmr tagging. In *Proceedings of the second SIGHAN workshop on Chinese language processing* (Morristown, NJ, USA, 2003), Association for Computational Linguistics, pp. 176–179.

[19] Zhang, H., Liu, T., Ma, J., and Liao, X. Chinese word segmentation with multiple postprocessors in hit-ir lab. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing* (2005), pp. 172–175.

[20] Zhang, R., Kikui, G., and Sumita, E. Subword-based tagging by conditional random fields for chinese word segmentation. In *NAACL '06: Proceedings of the Human Language Technology Conference of the NAACL* (Morristown, NJ, USA, 2006), Association for Computational Linguistics, pp. 193–196.

[21] Zhang, Y., and Clark, S. Chinese segmentation with a word-based perceptron algorithm. In *ACL '07: Proceedings of the 45nd Annual Meeting on Association for Computational Linguistics* (2007), Association for Computational Linguistics, pp. 840–847.

[22] Zhao, H., Huang, C.-N., Li, M., and Lu, B.-L. A unified character-based tagging framework for chinese word segmentation. *ACM Transactions on Asian Language Information Processing (TALIP) 9*, 2 (2010), 1–32.

Biographical Note

Yongshun Chen was born in October 1981, in Beijing, China. She received her Bachelor of Engineering degree in Telecommunication Engineering from Beijing University of Technology in July 2004. She joined Oregon Health & Science University (OHSU) in the Fall of 2008. She was then exposed to various topics in the broad area of speech and natural language processing.