Improving Chinese Word Segmentation and POS Tagging with Semi-supervised Methods Using Large Auto-Analyzed Data

Yiou Wang†, Jun’ichi Kazama†, Yoshimasa Tsuruoka‡†, Wenliang Chen§†, Yujie Zhang∗†, Kentaro Torisawa†

†National Institute of Information and Communications Technology (NICT), Japan
‡School of Information Science, JAIST, Japan; ∗Beijing Jiaotong University, China
§Human Language Technology, Institute for Infocomm Research, Singapore

{wangyiou, kazama, torisawa}@nict.go.jp; tsuruoka@jaist.ac.jp
wechen@i2r.a-star.edu.sg; yjzhang@bjtu.edu.cn

Abstract

This paper presents a simple yet effective semi-supervised method to improve Chinese word segmentation and POS tagging. We introduce novel features derived from large auto-analyzed data to enhance a simple pipelined system. The auto-analyzed data are generated from unlabeled data by using a baseline system. We evaluate the usefulness of our approach in a series of experiments on Penn Chinese Treebanks and show that the new features provide substantial performance gains in all experiments. Furthermore, the results of our proposed method are superior to the best reported results in the literature.

1 Introduction

In Chinese, most language processing starts from word segmentation and part-of-speech (POS) tagging. These two steps tokenize a sequence of characters without delimiters into words and predict a syntactic label (POS tag) for each segmented word. They are considered indispensable steps for higher-level NLP tasks such as parsing and information extraction. Although the performance of Chinese word segmentation and POS tagging has been greatly improved over the past years, the task is still challenging.

To improve the accuracy of NLP systems, one of the current trends is semi-supervised learning, which utilizes large unlabeled data in supervised learning. Several studies have demonstrated that the use of unlabeled data can improve the performance of NLP tasks, such as text chunking (Ando and Zhang, 2005), POS tagging and named entity recognition (Suzuki and Isozaki, 2008), and parsing (Suzuki et al., 2009; Chen et al., 2009; Koo et al., 2008). Therefore, it is attractive to consider adopting semi-supervised learning in Chinese word segmentation and POS tagging tasks.

In this paper, we present an approach to improve the performance of both segmentation and POS tagging by incorporating large unlabeled data. We first preprocess unlabeled data with our baseline models. We then extract various items of dictionary information from the auto-analyzed data. Finally, we generate new features that incorporate the extracted information for both word segmentation and POS tagging. We also perform word clustering on the auto-segmented data and use word clusters as features in POS tagging. In addition, we introduce lexicon features by using a cross-validation technique.

The use of sub-structures from the auto-annotated data has been presented previously (Noord, 2007; Chen et al., 2008; Chen et al., 2009). Chen et al. (2009) extracted different types of subtrees from the auto-parsed data and used them as new features in standard learning methods. They showed this simple method greatly improves the accuracy of dependency parsing. The idea of combining word clusters with discriminative learning has been previously reported in the context of named entity recognition (Miller et al., 2004; Kazama and Torisawa, 2008) and dependency parsing (Koo et al., 2008). We adapt and extend these techniques to Chinese word segmentation and POS tagging, and demonstrate their effectiveness in this task.

One of our criteria in this study was to achieve high accuracy with simple and easy-to-implement techniques. To meet this, the whole system is a pipeline with a character-based CRF for word segmentation and a word-based CRF for POS tagging. The information of unlabeled data is incorporated as additional new features without changing the learning algorithm.

To demonstrate the effectiveness of our approach, we conduct segmentation and POS tagging experiments on three versions of Penn Chinese Treebank, including the newly released C TB
Table 1: Word representation with a 6-tag tagset: S, B, B₂, B₃, M, E

| Type                  | Feature          | Description                                      |
|-----------------------|------------------|--------------------------------------------------|
| Character Unigram     | C_{i-1}, C_i, C_{i+1} | Previous, current and next character             |
| Nearing Character Bigram | (C_{i-1}, C_i), (C_i, C_{i+1}) | Previous (next) character and current character |
| Jump Character Bigram | C_{i-1}, C_i     | Previous character and next character            |
| Punctuation           | IsPu(C_0)        | Current character is punctuation                 |
| Character Type        | K(C_{i-2})K(C_{i-1})K(C_i)K(C_{i+1})K(C_{i+2}) | Types of character: date, numeral, alphabet, Chinese |

Table 2: Feature templates for word segmentation

version 7.0. We show that our semi-supervised approach yields improvements for all the test collections and achieves better results than the best reported results in the literature.

2 Segmentation and POS tagging Models

We implement our approach using sequential tagging models. Following the previous work (Zhao et al., 2006; Zhao et al., 2010), we employ the linear chain CRFs (Lafferty et al., 2001) as our learning model. Specifically, we use its CRF++ (version 0.54) implementation by Taku Kudo.

2.1 Baseline Segmentation Model

We employ character-based sequence labeling for word segmentation. In addition to its simplicity, the advantage of a character-based model is its robustness to the unknown word problem (Xue, 2003). In a character-based Chinese word segmentation task, a character in a given sequence is labeled by a tag that stands for its position in the word that the character belongs to. Zhao et al. (2006) reported that a 6-tag tagset shown in Table 1 is the best choice among the tagsets tested for Chinese word segmentation under the CRF framework. Therefore we also use this 6-tag tagset.

The basic types of features of our word segmentation model are listed in Table 2. These basic feature templates are based on Zhao et al. (2006; 2010) and Low et al. (2005).

2.2 Baseline POS Tagging Model

Since we employ a pipelined method, the POS tagging can be performed as a word labeling task, where the input is a sequence of segmented words. We use a CRF here as well. The feature set of our baseline POS tagger, is listed in Table 3. These are adopted from Wu et al. (2008).

3 Our New Features

In this section, we describe our approach of effectively integrating useful information from unlabeled (and labeled) data into the above baseline models through features. We preprocess unlabeled data with our baseline models and obtain word-segmented sentences with POS tags, and generate new features from the auto-analyzed data. Although the focus of the paper is semi-supervised learning, we also extract a lexicon from the training corpus and use it to generate features. Figure 1 shows an overview of our approach. The rest of this section describes our features in detail.

3.1 New features for Word Segmentation

3.1.1 Semi-supervised n-gram features

In this section, we describe our approach of extracting character-level n-gram lists and generating n-gram features from unlabeled data. We followed the method of Chen et al. (2009), and modified the method for word segmentation and POS tagging. First, we preprocess unlabeled data using the baseline segmenter and obtain auto-segmented data. We then extract character n-gram lists from auto-segmented sentences. Finally, we generate n-gram features for word segmentation.

By using the baseline segmenter, each character \( c_i \) in the unlabeled data is labeled with a tag \( t_i \). In other words, the output of auto-segmentation is a sequence \( \{(c_i, t_i)\}_{i=1}^{L} \). Let \( g \) be a character n-gram (e.g., uni-gram \( c_i \), bi-gram \( c_i c_{i+1} \), tri-gram \( c_{i-1} c_i c_{i+1} \) and so on)\(^2\), and \( \text{seg} \) be a segmentation profile for n-gram \( g \) observed at each position. The segmentation profile can be tag \( t_i \) or the combination of tags. Take a bi-gram for example, \( \text{seg} \) may be \( t_i \) or \( t_i t_{i+1} \). Then,

\(^2\)Note that there are several alternative ways for extracting n-grams at position \( i \), for example \( c_{i-1} c_i \) for a bi-gram. In this paper, we used the way as explained here.
we can extract a list of \( \{(g, \text{seg}, f(g, \text{seg}))\} \) from the auto-segmented data. Here, \( f(g, \text{seg}) \) is the frequency of the cases where \( n \)-gram \( g \) is segmented with the segmentation profile \( \text{seg} \). Then, following Chen et al. (2009), we group entries in this list into three sets: high-frequency (HF), middle-frequency (MF), and low-frequency (LF). The sets are defined as follows: if \((g, \text{seg})\) is one of the top 5% most frequent entries, it is labeled as HF; if it is between top 5% and 20%, it is labeled as MF, otherwise it is labeled as LF. Finally the list can be transformed as a \( n \)-gram list \( L_{ng} = \{(g, \text{seg}, FL(g, \text{seg}))\} \), with \( FL(g, \text{seg}) \) being the frequency label determined as above.

We attempted to encode the information of the above \( n \)-gram list into a new type of features, called \( n \)-gram features. We tried several feature representations and generation methods and found that the feature derived from the bi-gram list with \( \text{seg} = t_i \) was most effective.

We generate the feature for the current character \( c_0 \) as follows. We retrieve a set of entries, whose \( g \) part matches the bi-gram \( c_0c_1 \), from \( L_{ng} \). Let this set be \( L_m \). From an entry in \( L_m \), we generate a feature string represented by

(a) \( \text{seg} = FL(g, \text{seg}) \)

Then, we concatenate the feature strings of all the entries in \( L_m \) as one \( n \)-gram feature. If there is no entry in \( L_m \), the feature representation is "ND".

For example, consider that \( L_m \) is \{ (幸(Xing))/福(Fu), B, HF), (幸(Xing))/福(Fu), B2, MF), (幸(Xing))/福(Fu), E, LF) \} and we are processing \( c_kc_{k+1} = "\text{幸}(Xing)/福(Fu)" \); consequently, the \( n \)-gram feature of \( c_k \) is represented as "B-HF|B2-MF|E-LF". Note that the concatenation is in lexicographic order of the feature strings for standardization.

### 3.1.2 Lexicon features

Although a character-based model is simple and robust to unknown words, a limitation is its inability to consider word-level information. If a sequence of characters matches a word in an existing dictionary, it may be a clue that the character sequence should be segmented as one word. Several studies showed that using a dictionary brings improvement for Chinese word segmentation (Low et al., 2005; Zhao et al., 2010). For a corpus-based word segmenter, a manually annotated corpus is essential. Thus we can easily compile a lexicon from a training corpus. We refer to the features related to this lexicon as lexicon features.

In this study, we extract a lexicon in the following way. We collect words and all possible POS tags of the words from the training corpus. For instance, for the word "交流(JiaoLiu)", the collected entry may be (交流(JiaoLiu), NN-VV). Here, "NN-VV" is a concatenation of all the observed POS tags. POS tags are in lexicographical order, as in "NN-VV". However, we were concerned that a lexicon compiled in this way could cause an overfitting problem and that meaningful weights for the lexicon features may not be learned. This concern was indeed confirmed by the preliminary experiments using the development set. To solve this problem, we used the following method to build and use lexicons. The method is based on the idea
of cross-validation.

- Divide the training corpus into ten equal-sized sets, as in the data preparation for 10-fold cross-validation.
- For each set, we compile a lexicon using the remaining nine sets and use this lexicon to generate features for this set.
- For the development and test sets, we collect a lexicon using the entire training corpus and use it for feature generation.

Because the lexicon is extracted from other sets, the weights for this feature will not be overestimated by the learning algorithm. This kind of cross-validation-like techniques are used in studies such as Collins (2002) and Martins et al. (2008) to avoid over-fitting to the training data. Our method can be considered as its application to lexicon extraction.

Using the extracted lexicon, we generate lexicon features as follows. If a character sequence starting with character \( c_0 \) matches some words in the lexicon, we greedily choose the longest such matching word \( w \). Letting \( LENV(w) \) be the length (the number of characters) of \( w \), we add the following feature for each character \( c_k \) in \( c_0, c_1, \ldots, c_{LEN(w)} \):

(b) \( P(c_k)/LEN(w) \cdot POSs(w) \)

Here, \( P(c_k) \) is the position number (i.e., \( k \)) of the character \( c_k \) in \( w \) and \( POSs(w) \) represents the POS tags of \( w \) in the lexicon. After generating these features, we increment the current position by \( LENV(w) \). If there is no matching word, we proceed to the next character. That is, the forward maximum matching is used.

For example, consider that the current character sequence \( c_0c_1c_2 = "\text{幸}(Xing)/\text{福}(Fu)" \) was matched with a lexicon entry (幸福(XingFu), JJ-NN-VA), the feature for \( c_0 \) "幸(Xing)" is "1/2-JJ-NN-VA" and the feature for \( c_1 \) "福(Fu)" is "2/2-JJ-NN-VA".

Several feature representations have been attempted: (i) using only position information, (ii) representing the position information in a 6-tag or 4-tag tagset, or (iii) representing only one POS tag with the highest frequency. Development experiments showed that the current encoding is more effective than others in word segmentation tasks.

Note that our lexicon feature uses POS tag information for word segmentation. The fact that this feature is very effective as reported in Section 4.3 is interesting, since this can be considered as "loose" information feedback from the later process. Although we need a POS tagged corpus even for segmentation, this will not be a big problem since we usually perform POS tagging as well in many applications.

### 3.2 New Features for POS Tagging

We generate \( n \)-gram and lexicon features for POS tagging as well. In addition, the features that incorporate word clusters derived from a large automatically analyzed corpus (referred to as cluster features) are introduced.

#### 3.2.1 Semi-supervised \( n \)-gram features

We preprocess auto-segmented data using the baseline POS tagger and can generate word-level \( n \)-gram lists \( L_{wg} = \{w, pos, FL(w, pos)\} \). Here, \( w \) is a word \( n \)-gram and \( pos \) is the POS tagging profile of the word \( n \)-gram. Different from segmentation, features generated from the word unigram list yielded the best results.

A feature of this type for the current word \( w_0 \) is generated as follows. We retrieve a set of entries, whose \( w \) part matches the uni-gram \( w_0 \), from \( L_{wg} \). Let this set be \( L_m \). In the error analysis, we found that some words were associated with several odd POS tags in the uni-gram list. For instance, in addition to (研究(YanJiu), NN, HF) and (研究(YanJiu), VV, HF), (研究(YanJiu), VA, LF) and (研究(YanJiu), CD, LF) may appear as entries in the word unigram list, due to mis-tagging by the baseline POS-tagger. Therefore we further impose a restriction based on the frequency as follows: if the number of entries with a \( HF \) label \( \geq \) threshold, only the entries with \( HF \) will be selected, and if the sum of entries with a \( HF \) or \( MF \) label \( \geq \) threshold, the entries with either \( HF \) or \( MF \) will be selected, otherwise, all of the entries in \( L_m \) will be selected. Here the threshold is set to 2 based on the development experiments. Let these selected entries be \( L_s \). From an entry in \( L_s \), we generate a feature string represented by

(c) \( pos - FL(w, pos) \).

Then, we concatenate the feature strings of all entries in \( L_s \) as one \( n \)-gram feature. As for the previous instance, the feature for "研究(YanJiu)" is encoded as "NN-HFVW-\( HF\)".

#### 3.2.2 Semi-supervised cluster features

Following the work of Koo et al. (2008), we produced the clusters of various levels of granularity,
We conducted word segmentation and POS tagging experiments on Penn Chinese Treebanks incorporating up to 200-million-word unlabeled data.

### 4.1 Data Set

To compare with previous studies, we selected the widely used CTB5 (LDC2005T01), and defined the training, development and test sets according to Kruengkrai (2009a). In order to increase the reliability of our findings, we also used CTB6 (LDC2007T36) and CTB7 (LDC2010T07), which are larger than CTB5. For CTB6, we used the same data split as recommended in the CTB6 document. Because CTB7 includes data from various sources and various genres, we made a new data split according to the following criteria:

- Put the test set and the development set data described in CTB7 documents into each data set.
- Put the test set and the development set data of CTB5 into each set.
- Put all double checked files into the test-set.
- Keep the data of different genres and sources in balance.
- Increase the size of the development and test sets to make the evaluation more reliable.

The test set and development set of the CTB7 data split used in this paper are detailed in Table 4, and we used the rest as the training set.

### 3.2.3 Lexicon features

We use the same lexicon extracted for word segmentation for POS tagging. We add the following feature for the current word \( w_0 \):

\[(e) \text{POS}(w_0)\]

Here, \( \text{POS}(w_0) \) are all possible POS tags of the current word \( w_0 \) in the lexicon. We also tried to use different lexicons, as well as representing the feature with only one POS tag with the highest frequency. However, the experimental results were not better than those by using the above simple method.

---

1. We used the word clustering tool, available from http://www.cs.berkeley.edu/~pliang/software/brown-cluster-1.2.zip, to produce word clusters.
2. This is the same as the CTB6 data split.
3. In CTB7, sentences checked twice are marked, and they are expected to have higher annotation quality.
4. CTB5 and CTB6 data splits include small development and test sets.
The words in the test set that are not in the training set (Sproat and Emerson, 2003). The development sets were used to obtain the optimal values of tunable parameters and feature configurations.

The unlabeled data for our experiments were taken from the XIN_CMN portion of Chinese Gigaword Version 2.0 (LDC2009T14), which has approximately 311 million words. Some of CTB data and Chinese Gigaword data are from the same source: Xinhua newswire between 1994 and 1998. In order to avoid overlap between the CTB data and the unlabeled data, we used only the articles published in 1991-1993 and 1999-2004 as unlabeled data, with 204 million words.\(^8\) Note that we only used one million words from this data for word clustering, because the clustering process is time-consuming and the amount is enough to show the impact of cluster feature.

### 4.2 Parameter Tuning

CRF++ has four major tunable parameters to control the training condition: \(a\), the regularization algorithm; \(c\), the balance between over-fitting and under-fitting; \(f\), the cut-off threshold for the feature frequencies; and \(p\), the number of threads. We used \(a = \frac{CRF-L2}{2}\) (Gaussian regularization) and \(f = 1\). We set \(p\) to 12 for all experiments to speed up the training. For the baseline segmentation model, we varied \(c\) in the range of \([1.0, 5.0]\) and found that setting \(c = 4.0\) yielded the best performance on the development set of CTB7. For our approach, we varied \(c\) in the range of \([0.3, 5.0]\) and found that setting \(c = 1.0\) yielded the best performance. For the POS tagging model, \(c\) was set to 4.0 in all of the methods. For the clustering tool, \(c\) (the number of clusters) was set to 1000.

### 4.3 Experimental Results

We evaluated both word segmentation (Seg) and joint word segmentation and POS tagging (Seg &Tag). We used recall (R), precision (P) and \(F_1\) as evaluation metrics.

The experimental results of word segmentation on CTB5, CTB6 and CTB7 test sets are shown in Table 7, where (a) refers to the \(n\)-gram feature generated from the unlabeled data and (b) refers to the lexicon feature. The results show that the \(n\)-gram feature was very effective in all experiments and that the combination of (a) and (b) can provide further improvement.

The experimental results of segmentation and POS tagging on CTB5, CTB6 and CTB7 test sets are shown in Table 8 and Table 9. Table 8 shows the results when we used the baseline segmenta-

---

### Table 6: Statistics of CTB5, CTB6 and CTB7 data splits

| Method | CTB5 | CTB6 | CTB7 |
|--------|------|------|------|
| \# of sent. training | 18,089 | 23,420 | 31,131 |
| \# of sent. dev | 350 | 2,079 | 10,136 |
| OOV rate (word) | 0.0454 | 0.0345 | 0.0346 |
| OOV rate (word & tag) | 0.0635 | 0.0634 | 0.0608 |

### Table 7: Results of word segmentation

| POS tag method | CTB5 | CTB6 | CTB7 |
|----------------|------|------|------|
| Baseline | 0.9318 | 0.9095 | 0.8937 |
| +(c) \(n\)-gram | 0.9333 | 0.9014 | 0.8938 |
| +(d) cluster | 0.9350 | 0.9026 | 0.8959 |
| +(e) lexicon | 0.9346 | 0.9015 | 0.8959 |
| +(c)+(d)+(e) | 0.9359 | 0.9048 | 0.8985 |

### Table 8: \(F_1\) results of segmentation and POS tagging (baseline model for word segmentation)

### Table 9: \(F_1\) results of segmentation and POS tagging (our best model for word segmentation)
Table 10: Comparison with previous studies on CTB5

| Method | CTB6 | CTB7 |
|--------|------|------|
| Ours   | 0.9811 | 0.9418 |
| Baseline | 0.9753 | 0.9318 |
| Z&C 10 | 0.9778 | 0.9367 |
| K 09a | 0.9787 | 0.9367 |
| K 09b | 0.9798 | 0.9400 |
| Jiang 08a | 0.9785 | 0.9341 |
| Jiang 08b | 0.9774 | 0.9337 |
| N&U 07 | 0.9796 | 0.9338 |

Table 11: Comparison with previous studies on CTB6 and CTB7

| Method | CTB6 | CTB7 |
|--------|------|------|
| Ours   | 0.9579 | 0.9112 | 0.9565 | 0.9046 |
| Baseline | 0.9513 | 0.8999 | 0.9408 | 0.8937 |
| K 09a | 0.9550 | 0.9050 | 0.9540 | 0.8986 |
| K 09b | 0.9551 | 0.9053 | 0.9546 | 0.8990 |

Table 12: $F_1$ Results comparison on development set

| Method   | Seg  | Seg&Tag | Seg  | Seg&Tag | Seg  | Seg&Tag |
|----------|------|---------|------|---------|------|---------|
| Ours     | 0.9628 | 0.9316 | 0.9619 | 0.9138 | 0.9536 | 0.9027 |
| Baseline | 0.9493 | 0.8934 | 0.9564 | 0.9052 | 0.9493 | 0.8934 |
| K 09b    | 0.9628 | 0.9291 | 0.9577 | 0.9063 | 0.9547 | 0.8989 |
| K 09a    | 0.9642 | 0.9288 | 0.9574 | 0.9061 | 0.9533 | 0.8984 |

Table 13: Results of McNemar’s test.

| Models          | CTB5 | p-value | CTB6 | p-value | CTB7 | p-value |
|-----------------|------|---------|------|---------|------|---------|
| Ours vs. K 09b(Seg) | 0.8054 | 5.0e-08 | ≈ 0  |         |      |         |
| Ours vs. K 09b(Seg&Tag) | 0.7060 | 1.6e-14 | ≈ 0  |         |      |         |
| Ours vs. Base(Seg) | 4.0e-06 | 1.8e-11 | ≈ 0  |         |      |         |
| Ours vs. Base(Seg&Tag) | 2.1e-06 | ≈ 0    | ≈ 0  |         |      |         |

4.4 Comparative Results

In this section, we compare our approach with the best previous approaches reported in the literature. The performance scores of previous studies are directly taken from their papers, except for N&U 07 (Nakagawa and Uchimoto, 2007), which is taken from Kruekgrai et al. (2009b). Z&C 10 refers to Zhang and Clark (2010). Two methods in Kruekgrai et al. (2009a; 2009b) are referred to as K 09a and K 09b. Jiang 08a and Jiang 08b refer to Jiang et al. (2008a; 2008b). Table 10 compares $F_1$ results on CTB5.0. The best score in each column is in boldface. The results of our approach are superior to those of previous studies for both Seg and Seg&Tag.

We also conducted experiments using the system implemented by Kruekgrai for comparison on CTB6 and CTB7 with two methods (K 09a and K 09b) and the $F_1$ results are shown in Table 11.

For reference, the results of the development set are also shown in Table 12. Although the Seg performances of CTB5 and CTB7 are lower than K 09a and K 09b, Seg&Tag achieves the best performance on all development sets.

4.5 Statistical Significance Tests

We evaluated statistical significance using McNemar’s test. With McNemar’ test, we compared the correctness of the labeling decisions of the two models. The null hypothesis is that the disagreements (correct vs. incorrect) are due to chance. For Seg, a word in the system output is considered correct if the word boundary is correctly identified. For Seg &Tag, a word is considered correct only when both the word boundary and its POS tag are correctly identified. Table 13 summarizes the results on test sets. These tests suggest that although the difference from K 09b for CTB5 data is not statistically significant, all other differences are clearly statistically significant ($p < 10^{-5}$).

4.6 Comparison with Self-Training

An alternative method of incorporating unlabeled data is self-training, so we also compared our results to the self-training method. Because no existing research was found concerning the self-training method on word segmentation and POS...
tagging for Chinese, we tested the simplest self-training here. We analyzed the unlabeled data with the baseline models, added the newly auto-labeled data to the training corpus, and trained a new model. Since the manually labeled data should be considered more important than the unlabeled data (McClosky et al., 2006), we also adjusted the weight of the labeled data to the integer in the range of [1, 5] in experiments. The results of all the experiments were not positive – we were not able to obtain any improvement over the baseline models in either word segmentation or POS tagging. Due to space limitation, we only include the results with the labeled data weight = 1. Other weights did not change the conclusion here. Table 14 shows the \( F_1 \) results on segmentation with different sizes of the additional data on the CTB7 test set. Table 15 shows the \( F_1 \) results on segmentation and POS tagging. The segmentation by the baseline model was used for all of the POS tagging experiments here.

### 5 Related Work

Our approach is to incorporate large unlabeled data in Chinese word segmentation and POS tagging.

For research using large unlabeled data, Suzuki and Isozaki (2008) and Suzuki et al. (2009) proposed semi-supervised learning algorithms on giga-word-scale unlabeled data and showed performance improvement in POS tagging, syntactic chunking, and named entity recognition. Instead of using specialized semi-supervised learning algorithms, Chen et al. (2009) used features based on sub-structures in auto-parsed data and demonstrated the effectiveness of these features. Koo et al. (2008) presented the use of cluster features. The advantage of the methods by Chen et al. (2009) and Koo et al. (2008) is their simplicity and flexibility. Our research applied these techniques to word segmentation and POS tagging rather than dependency parsing.

Yu et al. (2007) proposed a character-based joint method for word segmentation and POS tagging, in which they introduced an unsupervised method for unknown word learning. However, they only learned the unknown words from the test set. Zhao and Kit (2007; 2008) proposed an approach using unsupervised segmentation criteria as features for Chinese word segmentation. However, their features were only accumulated from the training and test data. Our approach differs in that we used features generated from large unlabeled data and provided richer information, which may be unseen from the training corpus.

Kruengkrai et al. (2009a; 2009b) presented a discriminative word-character hybrid model for joint Chinese word segmentation and POS tagging and achieved the state-of-the-art accuracy for the CTB test sets. Instead of using the hybrid model, we used conceptually simpler pipelined models built with standard CRF tools. Compared with their method, our approach achieved higher performance with the help of unlabeled data.

### 6 Conclusion

In this paper, we presented a simple yet effective semi-supervised approach to pipelined Chinese segmentation and POS tagging. Through a series of experiments, we demonstrated that our approach provides substantial improvement over the best previously reported methods as well as the baseline methods.

### References

Andr F. T. Martins, Dipanjan Das, Noah A. Smith, and Eric P. Xing. 2008. Stacking Dependency Parsers. In Proceedings of EMNLP-2008, pages 513-521.

Canasai Kruengkrai, Kiyotaka Uchimoto, Jun’ichi Kazama, You Wang, 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.
