Abstract

Word segmentation is usually recognized as the first step for many Chinese natural language processing tasks, yet its impact on these subsequent tasks is relatively under-studied. For example, how to solve the mismatch problem when applying an existing word segmenter to new data? Does a better word segmenter yield a better subsequent NLP task performance? In this work, we conduct an initial attempt to answer these questions on two related subsequent tasks: semantic slot filling in spoken language understanding and named entity recognition. We propose three techniques to solve the mismatch problem: using word segmentation outputs as additional features, adaptation with partial-learning and taking advantage of n-best word segmentation list. Experimental results demonstrate the effectiveness of these techniques for both tasks and we achieve an error reduction of about 11% for spoken language understanding and 24% for named entity recognition over the baseline systems.

1 Introduction

Unlike English text in which sentences are sequences of words separated by white spaces, in Chinese text (as are some other languages including Arabic, Japanese, etc.), sentences are represented as strings of characters without similar natural delimiters. Therefore, it is generally claimed that the first step in a Chinese language processing task is to identify the sequence of words in a sentence and mark boundaries in appropriate places, which is referred to as the task of Chinese Word Segmentation (CWS).

(1) Input: 能穿多少穿多少
CWS 1: 能|穿|多|多|穿|多|穿|多|穿|少|少|少|穿|少|少|少|穿|少
(Put on as much clothes as possible.)
CWS 2: 能|穿|多|少|穿|多|穿|少
(Put on as little clothes as possible.)

Word segmentations in Chinese text do reduce ambiguities. In the example (1), the same span of text (the input) can convey entirely opposite meanings (the English sentences in parentheses) depending on how word boundaries (CWS 1 and CWS 2) are labeled. Therefore, it is generally believed that more accurate word segmentations should benefit more the subsequent Chinese language processing tasks, such as part-of-speech tagging, named entity recognition, etc. There has been quite a number of research in the field of CWS to improve segmentation accuracy, yet its impact on the subsequent processing is relatively under-studied. Chang et al. (2008) explore how word segmentation improves machine translation; and Ni and Leung (2014) explore how word segmentation impacts automatic speech recognition yet do not have conclusive findings. In this research, we aim to better understand how CWS benefits the subsequent NLP tasks, using semantic slot filling in spoken language understanding (SLU) and named entity recognition (NER) as two case studies.

In particular, we investigate the impact of Chinese word segmentation in three different situations.

---

*Work done at Nuance during an internship.

1We use ‘|’ to indicate a word boundary. Example is borrowed and revised from (Chen et al., 2015).
First, assuming domain data (the data for a particular subsequent task, e.g., SLU or NER) having no word boundary annotation (§4), we can apply word segmenters trained with publicly-available data to the domain data to get the word boundary. However, existing word segmenters may have a domain mismatch problem due to the fact that they may have different genre from the subsequent task and are usually segmented with different standards (Huang and Zhao, 2007). Therefore, we propose three techniques to solve this problem. Note, these techniques can be used together.

1) We use word segmentation outputs as additional features in subsequent tasks (§3.2), which is more robust against error propagation than using segmented word units.

2) We adapt existing word segmenters with partially-labeled data derived from the subsequent task training data (§3.3), further improving the end-to-end performance.

3) We take advantage of the n-best list of word segmentation outputs (§3.4), making the subsequent task less sensitive to word segmentation errors.

Second, assuming domain training data (e.g., NER) is already segmented with word boundary (§5), we are able to train a domain word segmenter with the data itself and apply it to the testing data. This allows us to see the differences between a word segmenter trained with in-domain data and one trained with publicly-available data.

Last, assuming both domain training and testing data have word boundary information (§5), it allows to explore the upper bound performance of the subsequent task with a perfect word segmenter.

Experimental results show that the proposed techniques do improve the end-to-end performance and we achieve an error rate reduction of 11% for SLU and 24% for NER over their corresponding baseline systems. In addition, we found that even a word segmenter that is only moderately reliable is still able to improve the end-to-end performance, and a word segmenter trained with in-domain data is not necessarily better compared to a word segmenter trained with out-domain data in terms of the end-to-end performance.

2 Related Work

Word segmentation has received steady attention over the past two decades. People have shown that models trained with limited text can have a reasonable accuracy (Li and Sun, 2009; Zhang et al., 2013a; Li et al., 2013; Cheng et al., 2015). However, the fact is that none of existing algorithms is robust enough to reliably segment unfamiliar types of texts without fine-tuning (Huang et al., 2007). Several approaches have proposed to eliminate this issue, for example the use of unlabeled data (Sun and Xu, 2011; Wang et al., 2011; Zhang et al., 2013b) and partially-labeled data (Yang and Vozila, 2014; Takahasi and Mori, 2015). In our work, we encounter the same issue when applying word segmentation to the subsequent tasks and thus we propose three approaches to solve this problem.

Word segmentation has been applied in several subsequent tasks, e.g. NER (Zhai et al., 2004), information retrieval (Peng et al., 2002), automatic speech recognition (Ni and Leung, 2014), machine translation (Xu et al., 2008; Chang et al., 2008; Zhang et al., 2008; Zeng et al., 2014), etc. In general, there are two types of approaches to utilize word segmentation in subsequent tasks: pipeline and joint-learning. The pipeline approach creates word segmentation first and then feeds the segmented words into subsequent task(s). It is straightforward, but suffers from error propagation since an incorrect word segmentation would cause an error in the subsequent task. The joint-learning approach trains a model to learn both word segmentation and the subsequent task(s) at the same time. A number of subsequent tasks have been unified into joint models, including disambiguation (Wang et al., 2012), part-of-speech tagging (Jiang et al., 2008a; Jiang et al., 2008b; Zhang and Clark, 2010; Sun, 2011), NER (Gao et al., 2005; Xu et al., 2014; Peng and Dredze, 2015), and parsing (Hatori et al., 2012; Qian and Liu, 2012). However, the joint-learning process generally assumes the availability of manual word segmentations for the training data, which limits the use of this approach. Thus in this work, we focus on the pipeline approach, but instead of feeding the segmented words, we use word segmentation results as additional features in the subsequent tasks, which is more robust against error propagation.
3 Applying CWS to Subsequent Tasks

In this section, we describe how to integrate word segmentation information when domain data having no word boundary information, using SLU and NER as two case studies.

We first introduce the baseline system, and then describe the techniques that we propose to solve the domain mismatch problem when applying automatic CWS to the subsequent NLP tasks.

3.1 Baseline system

Both of the SLU and NER can be formulated as sequence labeling tasks, and can be solved using machine learning techniques such as Conditional Random Field (CRF), Recurrent Neural Network, or their combinations (Wan et al., 2011; Mesnil et al., 2015; Rondeau and Su, 2015). We adopt the tool wapiti (Lavergne et al., 2010), which is an implementation of CRF. In the baseline system, each Chinese character is treated as a labeling unit. Here is an example of our training sentences for SLU:

```
三|division|元|division|里|division|莫|street|干|street|山|street|路|street|周|locref|围|locref|的|unk|餐|query|厅|query'
```

(Find the restaurants near Sanyuanli Mogan Mountain road). The input features for training the baseline CRF model are character ngrams in the K-window and label bigrams. For computational efficiency, we use trigram within 5-character window. Given the current character \( c_0 \), we extract the following character ngram features:

- \( c_2 \), \( c_1 \), \( c_0 \), \( c_1 \), \( c_2 \), \( c_2 \), \( c_2 \), \( c_1 \), \( c_2 \), \( c_1 \), \( c_0 \), \( c_1 \), \( c_2 \), \( c_1 \), \( c_0 \).

3.2 Using CWS as features

When word segmentation information is not available within the domain data, we can use publicly-available corpora such as the Chinese Tree Bank (Levy and Manning, 2003), to train an automatic word segmenter.

A dominant approach for supervised CWS is to formulate it as a character sequence labeling problem, and label each character with its location in a word (Xue, 2003). A popular labeling scheme is ‘BIES’: ‘B’ for the beginning character of a word, ‘I’ for the internal characters, ‘E’ for the ending character, and ‘S’ for single-character word. Following (Yang and Vozila, 2014), we train our automatic word segmenter with CRF using the input features of character unigrams and bigrams, consecutive character equivalence, separated character equivalence, punctuation, character sequence pattern, anchor of word unigram and bigram. This word segmenter achieves state-of-the-art or comparable performance.

A straightforward way to integrate word segmentation is the traditional pipeline approach. It uses word segmentation first and feeds the segmented words to subsequent task(s), named as **Word Unit**. However, this method suffers from the error propagation problem since an incorrect word segmentation would cause an error in the subsequent task. Therefore, we proposed to use word segmentation outputs as additional features (**As Features**) in the subsequent tasks, as introduced below. We hypothesize the **As Features** is less sensitive to word segmentation errors since the CRF model can still rely on the character features when a word segmentation is not perfect.

**Word Unit** We can use segmented words instead of characters as labeling units for the CRF learning. During training we can run forced-decoding (Lavergne et al., 2010) on word segmentation so that word boundaries are consistent with semantic slot or named entity boundaries. During testing we simply apply the word segmenter to the sentences.

**As Features** We can still keep using characters as the labeling units, but add the word segmentation information as additional features. Given the current character \( c_0 \) and word segmentation output represented as ‘BIES’ tag \( t_0 \), we extract the character ngram features together with the following word segmentation tag ngram features:

- \( t_2 \), \( t_1 \), \( t_0 \), \( t_1 \), \( t_2 \), \( t_2 \), \( t_1 \), \( t_1 \), \( t_0 \), \( t_1 \), \( t_1 \), \( t_0 \), \( t_1 \), \( t_0 \), \( t_1 \), \( t_0 \), \( t_1 \).

The tag ngram features provide word segmentation information indirectly. For example, \( t_0t_1='BE' \) indicates \( c_0 \) initiates a two-character word, while \( t_0t_1t_2='BII' \) means that \( c_0 \) is probably a beginning of a long word.

3.3 Adaptation with Partial-learning

The publicly-available corpora for word segmentation, however, may create a domain-mismatch prob-
Figure 1: Partially-labeled word segmentations derived from named entity labels. The first character in a name (‘美’ in the organization name ‘美中贸易全国委员会’) can only be labeled as ‘S’ or ‘B’, while the last one (‘会’) can only be labeled as ‘S’ or ‘E’; similarly, a character after a name (‘主’) can only be labeled as ‘S’ or ‘B’, while a character before a slot (‘席’) can only be labeled as ‘S’ or ‘E’.

Even if the NER/SLU task training data is not word segmented, the semantic slot and named entity labels actually provide valuable information on word boundaries. As illustrated in Fig. 1, the first character in an organization/person/location name can only be labeled as ‘S’ or ‘B’, while the last one can only be labeled as ‘S’ or ‘E’; similarly, a character after a name can only be labeled as ‘S’ or ‘B’, while a character before a name can only be labeled as ‘S’ or ‘E’. We can thus create partially-labeled CWS data from SLU and NER labels. These partially-labeled data can then be used to adapt the out-of-domain word segmenter trained from publicly-available corpus.

Täckström et al. (2013) propose the approach partial-label learning to learn from partially-labeled data, and Yang and Vozi (2014) apply it to Chinese word segmentation. In partial-label training, each item in the sequence receives multiple labels, and each sequence has a lattice constraint, as shown in Fig. 1. The basic idea is to marginalize the probability mass of the constrained lattice in a cost function. The marginal probability of the lattice is defined as Equation 1, where $C$ denotes the input character sequence, $L$ denotes the label sequence, and $\hat{Y}(C, \hat{L})$ denotes the constrained lattice (with regard to the input sequence $C$ and the partial-labels $\hat{L}$).

$$p_\theta(\hat{Y}(C, \hat{L})|C) = \sum_{L \in \hat{Y}(C, \hat{L})} p_\theta(L|C)$$

The optimization objective function is to maximize the log-likelihood of the training set, in which likelihood is calculated via the probability mass of the constrained lattice, as shown in Equation 2. Here $n$ denotes the number of sentences in the training set.

$$L(\theta) = \sum_{i=1}^{n} \log p_\theta(\hat{Y}(C_i, \hat{L}_i)|C_i)$$

With CRF\(^2\), a gradient-based approach such as L-BFGS can be used to optimize Equation 2. We expect that this adaptation process should help to provide better word segmentation information that further improves the subsequent task performance.

3.4 N-best CWS

Only using the best word segmentation output as features for the subsequent tasks might not be sufficient (as we will show in our experiments). Indeed we can make use of the n-best word segmentation outputs. The task of SLU or NER is to find the best label sequence $L$, given the character sequence $C$, represented as $\arg \max_{L} P(L|C)$. By including the word segmentation information, we can rewrite it by marginalizing over all possible word segmentations.

$$\arg \max_{L} P(L|C) = \arg \max_{L} \sum_{j} P(L, W_j|C)$$

We modified wapiti to implement partial learning.
Where, \( W_j \) is each possible word segmentation. This formula can be understood as two components: \( P(W_j | C) \) is the word segmentation model and \( P(L | W_j, C) \) is the SLU/NER model. In practice, we can use the n-best outputs associated posterior probabilities from the wapiti, for both \( P(W_j | C) \) and \( P(L | W_j, C) \).

### 4 CWS for SLU

In this section, we investigate the impact of CWS to the task of spoken language understanding (SLU) by making use of existing word segmenters trained with publicly-available data (1\textsuperscript{st} situation in §1). This is motivated by the fact that our SLU training and testing data are not pre-segmented by semantic word units.

We choose semantic slot filling in SLU because it is becoming popular as it is a critical component to support conversational virtual assistants, such as Apple Siri, Samsung S Voice, Microsoft Cortana, Nuance Nina, just to name a few. The task of SLU is to convert a user utterance into a machine-readable semantic representation, which typically includes two sub-tasks: intent recognition and semantic slot filling (Tur et al., 2013). Intent recognition is to determine the intention of the user utterance. For example, for the input utterance ‘book a ticket from Boston to Seattle’, SLU will determine that its intent is \textit{ticket-booking} as opposed to \textit{music-playing}. Semantic slot filling is to extract the designated slot values for the recognized intent from the input utterance. For example, SLU will extract ‘depart:Boston’ and ‘arrive:Seattle’ from the above user utterance. In this paper, we assume the availability and correctness of intent recognition, and focus only on semantic slot filling.

#### 4.1 SLU experiments setting

As described above, intent recognition is the first step in SLU, and the availability of which is assumed in this research work. We organize our training and testing data for semantic slot filling according to their intents. A single model for semantic slot filling is trained for each individual intent because different intents have different designated slots. For example, for the intent \textit{ticket-booking}, the designated slots are the arrival and departure city/airport, airline, date, etc.; While the \textit{local-search} intent is more interested in the city, address, street name, type of point of interest, etc. For evaluation, each model is applied to the corresponding intent’s testing data. At the end, we gather the automatic semantic labels of all intents in a pool and calculate F-measure.

Our SLU data consists of about 2 million sentences for training and 260 thousand sentences for testing, distributing into 170 intents.

#### 4.2 Results and discussion

We build two word segmenters from two public corpora, the Chinese Tree Bank 6 (CTB6) and the PKU corpus from the SIGHAN Bakeoff 2005, respectively. The data statistics of the two corpora are shown in Table 1.

<table>
<thead>
<tr>
<th></th>
<th>CTB6</th>
<th>PKU</th>
</tr>
</thead>
<tbody>
<tr>
<td>number of sentences</td>
<td>23,458</td>
<td>19,058</td>
</tr>
<tr>
<td>number of unique character</td>
<td>4,223</td>
<td>4,685</td>
</tr>
<tr>
<td>number of unique word</td>
<td>42,127</td>
<td>55,302</td>
</tr>
<tr>
<td>average sentence length</td>
<td>45.0</td>
<td>95.8</td>
</tr>
<tr>
<td>average word length</td>
<td>1.7</td>
<td>1.7</td>
</tr>
</tbody>
</table>

Table 1: Statistics of two publicly-available corpus for CWS training.

Baseline using only character ngram features gives an F-measure of 93.92%. When switching to using automatic segmented words as the labeling units (Word Unit), the performance is a lot worse in both cases (87.10% for CTB6 and 88.68% for PKU). This indeed is not too surprising because errors in CWS propagate into SLU semantic slot filling. If an error results in a word crossing the boundary of semantic slots, it will definitely lead to an error in SLU semantic slot filling.

On the other hand, when supplying the automatic ‘BIES’ ngrams from CWS to SLU semantic slot filling (As Features), we observe a nice gain in both cases, 94.41% for CTB6 and 94.13% for PKU. Using the ngram ‘BIES’ as input features provides useful information of word segmentation to SLU semantic slot filling, while it is less sensitive to word segmentation errors.
Table 2: SLU Results in Recall (R), Precision (P), and F-measure (F). * means it is statistically significant better than Baseline using a Z-test with a confidence level of 99%.

<table>
<thead>
<tr>
<th></th>
<th>CTB6</th>
<th></th>
<th>PKU</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R (%)</td>
<td>P (%)</td>
<td>F (%)</td>
</tr>
<tr>
<td>Baseline</td>
<td>94.10</td>
<td>93.73</td>
<td>93.92</td>
</tr>
<tr>
<td>Word Unit As Features</td>
<td>89.42</td>
<td>84.90</td>
<td>87.10</td>
</tr>
<tr>
<td>Partial Learning</td>
<td>94.18</td>
<td>94.76</td>
<td>94.47*</td>
</tr>
<tr>
<td>N-best</td>
<td>94.36</td>
<td>94.84</td>
<td>94.60*</td>
</tr>
</tbody>
</table>

(2) Input: 查找[湖南财经经济学院][附近][的][餐厅]4
CWS: 查找[湖南|财政经济学院][附近][的][餐厅]
(Find the restaurants near Hunan College of Finance and Economics)

Example (2) illustrates that how CWS helps SLU semantic slot filling. For the sentence, the baseline system extracts ‘湖南财’ as a location name. However, the word segmentation separates the words ‘湖南’ (Hunan) and ‘财政’ (Finance), which reduces the probability score of ‘湖南财’ being a slot value because it crosses word boundaries. With CWS information, the system is able to extract ‘湖南财政经济学院’ (Hunan College of Finance and Economics) as a slot value.

(3) Input: 转发[淘宝网的链接]
CWS 1: 转发[淘宝网的链接]
(Forward the link of taobao.com)
CWS 2: 转发[淘宝网的链接]
(Forward the link of taobao.com)

(4) Input: 亲[四季酒店]在哪里
CWS 1: 亲[四季酒店]在哪里
(Where is the Kiss Four Seasons Hotel)
CWS 2: 亲[四季酒店]在哪里
(Dear, where is the Four Seasons Hotel)

Adapting the word segmentation with SLU partially-labeled data gives further gain to semantic slot filling. In the case of CTB6 it reaches an F-measure of 94.47%, and 94.48% in PKU, using the ngram of ‘BIES’ labels from the adapted segmenters. Here are two examples showing how the adaptation process further improves SLU. In the example (3), we have the incorrect word segmentation (CWS 1) before adaptation. It splits a word ‘淘宝’ (taobao.com) and thus labels ‘宝网的链接’ as a semantic slot. From the adaptation the system learns that ‘淘宝网’ is a word, and it generates the correct word segmentation (CWS 2) and thus is able to create the correct semantic slot value ‘淘宝网的链接’ (the link of taobao.com). Similarly, in the example (4), the sentence is initially under-segmented (CWS 1) and it creates the incorrect semantic slot value ‘亲四酒店’. From the adaptation the system learns to put a word boundary between ‘亲’ and ‘四’ and then the correct slot value ‘四季酒店’ (Four Seasons Hotel) is extracted.

Finally, we take 10-best outputs from the adapted word segmenter, for each word segmentation generate 10-best SLU outputs, sum up the probabilities, and search for the best semantic label sequence following Equation 3. We further push the performance to an F-measure of 94.60% for CTB6 and 94.61% for PKU. Compared with the baseline system that uses character ngrams as input features, the information of CWS helps us achieve an error reduction of about 11%.

5 CWS for NER

In our experiments on SLU, we showed how CWS helps the subsequent task when no in-domain word segmentation data is available (1st situation in §1). In this section, we investigate the impact of CWS to another important subsequent task: named entity recognition (NER). For the NER data we use, both the domain training and testing data have word boundary information, which allows us to explore the differences between word segmenters trained with in-domain data and publicly-available data (2nd situation). It also allows us to see the performance of the subsequent task using manual word segment-
tation (3rd situation). Moreover, it allows us to see the relationship between the performance of word segmentation and the end-to-end subsequent task.

5.1 NER experiments setting

For NER experiments, we use the benchmark NER data from the third SIGHAN Chinese language processing Bakeoff (SIGHAN-3) (Levow, 2006). It consists of 46,364 sentences in the training set and 4,365 sentences in the testing set. These data are annotated with both word boundaries and NER information.

5.2 Results and discussion

**Baseline** system which only uses character ngram features (same configuration as the SLU task) gives the performance of 85.81% in F-measure, as shown in Table 3.5

**Oracle** system uses character ngram features together with manual in-domain word boundary information during both training and testing, showing that perfect word segmentation information does help NER a lot. Again this suggests that good word segmentation does reduce ambiguities for the subsequent NLP tasks, as we argue in the introduction. Of course, since manual word segmentation is not generally available (esp. on testing), this raises the motivation of our research work: what is the impact of automatic CWS on NER and how to make the best out of it.

To understand the impact of automatic CWS on NER, we discard the manual word segmentations in the NER data, and build two word segmenters from two public corpora, CTB6 and PKU respectively, same as we did for the SLU experiments. We also adapt them to NER with partial-label learning, and finally apply n-best CWS to NER decoding. Here we only report the results for As Features, as summarized in Table 3. Similar to SLU, when supplying the automatic ‘BIES’ ngrams from CWS to NER (As Features), we observe a nice gain in both cases of CTB6 and PKU. The NER F-measure improves to 86.40% and 87.05% respectively. In addition, adapting the word segmentation with NER partially-labeled data gives a further gain for both CTB6 and PKU, with an F-measure of 86.96% and 87.64% respectively. Note that, the adaptation process does improve the CWS performances for both CTB6 and PKU.

**In-domain CWS**

NER system uses the NER training data to build a word segmenter and then apply it to the NER training and testing data to extract the word segmentation features. A naive thought is that it will result in a better NER performance than CTB6 and PKU since a word segmenter trained with the in-domain data should be better than one trained with publicly-available data due to the domain mismatch issue. As shown in Table 3, it is true that the word segmentation F-measures of NER are much better than CTB6 and PKU. However, to our surprise, the NER F-measure is only 83.45%, which is even worse than Baseline.

We hypothesize that this is due to the mismatch of the training CWS and testing CWS (as shown in Table 3, CWS F (train) and F (test)). When CWS accuracy is high on the training data, the NER model trained with such data puts more weight on word segmentation features rather than character features. However, during testing, the performance of CWS drops, resulting in more word segmentation errors, with a high chance to propagate to NER errors; even worse, a lot of these CWS errors are around NERs since a lot of NERs are OOVs and thus are challenging to segment correctly. To test this hypothesis, we use 3-fold cross-validation to get the word boundary information during the CWS training, and thus the model is named as NER 3-fold. Note, although the performance of CWS decreases in the training, it has a more balanced CWS performance between training and testing, which gives a better NER performance (improving 83.45% from NER to 86.80%).

**N-best CWS**

The model N-best takes N-best outputs from the adapted word segmenter, for each word segmentation generate K-best NER outputs, sums up the probabilities and searches for the best named-entity
Table 3: CWS and NER Results in F-measure. **CWS F (Train)** and **CWS F (Test)** are the word segmentation F-measure in the training and testing data respectively. **NER F** is the named-entity testing F-measure. ‘-’ means that the metric does not apply. For example, **Baseline** has no word segmentation model and F-measure cannot be calculated for N-best models. For **N-best**, we set N=10 and K=2. ‘*’ means it is statistically significant better than **Baseline** using a Z-test with a confidence level of 99%.

<table>
<thead>
<tr>
<th></th>
<th>CWS F (Train) (%)</th>
<th>CWS F (Test) (%)</th>
<th>NER F (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Baseline</strong></td>
<td>-</td>
<td>-</td>
<td>81.63</td>
</tr>
<tr>
<td><strong>Oracle</strong></td>
<td>100</td>
<td>100</td>
<td>92.01</td>
</tr>
<tr>
<td><strong>CTB6</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>As Features</td>
<td>84.16</td>
<td>84.71</td>
<td>82.91</td>
</tr>
<tr>
<td>Partial Learning</td>
<td>85.21</td>
<td>85.21</td>
<td>83.78</td>
</tr>
<tr>
<td>N-best</td>
<td>-</td>
<td>-</td>
<td>86.88</td>
</tr>
<tr>
<td><strong>PKU</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>As Features</td>
<td>86.53</td>
<td>87.37</td>
<td>84.04</td>
</tr>
<tr>
<td>Partial Learning</td>
<td>87.56</td>
<td>87.57</td>
<td>84.81</td>
</tr>
<tr>
<td>N-best</td>
<td>-</td>
<td>-</td>
<td>87.44</td>
</tr>
<tr>
<td><strong>NER</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>As Features</td>
<td>99.64</td>
<td>95.70</td>
<td>80.88</td>
</tr>
<tr>
<td>N-best</td>
<td>-</td>
<td>-</td>
<td>84.55</td>
</tr>
<tr>
<td><strong>NER 3-fold</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>As Features</td>
<td>94.69</td>
<td>95.70</td>
<td>83.61</td>
</tr>
<tr>
<td>N-best</td>
<td>-</td>
<td>-</td>
<td>87.22</td>
</tr>
<tr>
<td><strong>SIGHAN-3 Best System</strong></td>
<td>-</td>
<td>-</td>
<td>84.20</td>
</tr>
</tbody>
</table>

**SIGHAN-3 evaluation**

In the closed track evaluation of SIGHAN-3 (Levow, 2006), participants could only use the information found in the provided training data. Our best model (**NER 3-fold**) belongs to this track since it uses only the word segmentation annotation in the training data set. Our model outperforms all the submissions as shown in Table 3. Furthermore, even if manual word segmentation does not exist in the data, the model **CTB6 N-best** and **PKU N-best** which using existing word segmenters trained from publicly-available data can still outperform all the submissions in SIGHAN-3. Note that, these models use only character and word segmentation features without requiring additional name lists, part-of-speech taggers, etc.

6 Conclusion and future work

Chinese word segmentation is an important research topic and usually is the first step in Chinese natural language processing, yet its impact on the subsequent processing is relatively under-studied. To our knowledge, this research work is the first attempt to understand in depth how automatic CWS impacts the two related subsequent tasks: SLU semantic slot filling and named entity recognition.

In this work, we proposed three techniques to
solve the domain mismatch problem when applying CWS to other tasks: using word segmentation outputs as additional features, adaptation with partial-learning and taking advantage of n-best list. All three techniques work for both tasks.

We also examined the impact of CWS in three different situations: First, when domain data has no word boundary information, we showed that a word segmenter built from public out-of-domain data is able to improve the end-to-end performance. In addition, adapting it with the partially-labeled data derived from human annotation can further improve the performance. Moreover, marginalizing n-best word segmentations leads to further improvement. Second, when domain word segmentation is available, the word segmenter trained with the domain data itself has a better CWS performance but it does not necessarily have a better end-to-end task performance. A word segmenter with more balanced performance on the training and testing data may obtain a better end-to-end performance. Third, when testing data is manually segmented, word segmentation does help the task a lot. This is not a typical use case in reality, but it does suggest that word segmentation does reduce ambiguities for the subsequent NLP tasks.

In the future, we can try to sequentially stack two CRFs (one for word segmentation and one of subsequent task). We also would like to explore more subsequent tasks beyond sequence labeling problems.

Acknowledgments

The authors would like to thank Paul Vozila, Xin Lu, Xi Chen, and Xiang Li for helpful discussions, and the anonymous reviewers for insightful comments.

References


Gina-Anne Levow. 2006. The third international Chinese language processing bakeoff: Word segmentation


