Discriminative Boosting from Dictionary and Raw Text – A Novel Approach to Build A Chinese Word Segmenter

Fandong Meng Wenbin Jiang Hao Xiong Qun Liu
Key Laboratory of Intelligent Information Processing
Institute of Computing Technology
Chinese Academy of Sciences
{mengfandong, jiangwenbin, xionghao, liuqun}@ict.ac.cn

ABSTRACT
Chinese word segmentation (CWS) is a basic and important task for Chinese information processing. Standard approaches to CWS treat it as a sequence labelling task. Without manually annotated corpora, these approaches are ineffective. When a dictionary is available, dictionary maximum matching (DMM) is a good alternative. However, its performance is far from perfect due to the poor ability on out-of-vocabulary (OOV) words recognition. In this paper, we propose a novel approach that integrates the advantages of discriminative training and DMM, to build a high quality word segmenter with only a dictionary and a raw text. Experiments in CWS on different domains show that, compared with DMM, our approach brings significant improvements in both the news domain and the Chinese medicine patent domain, with error reductions of 21.50% and 13.66%, respectively. Furthermore, our approach achieves recall rate increments of OOV words by 42.54% and 23.72%, respectively in both domains.

KEYWORDS: discriminative model, word segmentation, dictionary maximum matching.

Proceedings of COLING 2012: Posters, pages 819–828,
COLING 2012, Mumbai, December 2012.

1 Introduction

Word segmentation is a basic and important task for information processing of Chinese. Most effective approaches (Xue and Shen, 2003; Ng and Low, 2004) to CWS treat it as a character tagging task, in which the model used to make tagging decisions can be trained by discriminative methods, such as Maximum Entropy (ME) (Ratnaparkhi and Adwait, 1996), Conditional Random Fields (Lafferty et al., 2001), perceptron training algorithm (Collins, 2002; Collins and Roark, 2004), etc. These methods have achieved good results, but rely on large scale high quality annotated corpora, which are rare in resource-poor languages and domains. Besides, directly adapting a classifier trained on one domain to another domain leads to poorer performance [1]. Given a dictionary, dictionary maximum matching (DMM) is an alternative in the case of no available annotated corpora, but its performance is not satisfying due to the poor ability on OOV words recognition.

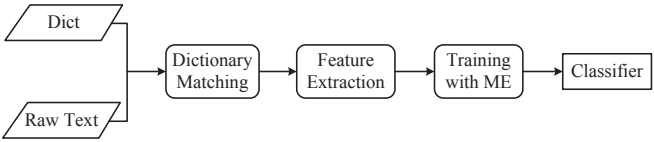

Figure 1: The pipeline of our method.

In this paper, we propose a novel approach of integrating the advantages of discriminative training and DMM, that enables us to utilize only a domain dictionary and a raw corpus to build a high quality word segmenter. Figure 1 describes the pipeline of our method. First, we scan each sentence in the raw corpus, and compare the continuous characters with the words in dictionary by reliable dictionary matching strategy. If successfully matched, we extract feature instances from the matching parts and construct reliable feature instance set. Finally, we train a classifier using all the reliable feature instances by Maximum Entropy approach.

To test the efficacy of our method, we do experiments in CWS, with Penn Chinese Treebank 5.0 (CTB) (Xue et al., 2005) and Chinese medicine patent (CMP) corpus. When we use the classifier trained on CTB to process the CMP testing set, the performance is poorer than DMM with dictionary of CMP. Besides, compared with DMM, our method achieves significant improvements with error reductions of 21.50% and 13.66%, respectively. Besides, the recall rate increments of OOV words are 42.54% and 23.72%, respectively (Section 4).

2 Segmentation as Gap Tagging Classification

Before describing the method of dictionary matching based feature instances extraction, we give a brief introduction of gap tagging classification strategy for segmentation. Formally, a Chinese sentence can be represented as a character sequence: $C_{1:n} = C_1C_2\cdots C_n$, where $C_i(i = 1,\cdots,n)$ is a character. We explicitly add the gap $G_i(i = 1,\cdots,n-1)$ of character C_i and C_{i+1} to the sentence $C_{1:n}$, denoted as $C_{1:n}|G_{1:n-1} = C_1G_1C_2G_2\cdots G_{n-1}C_n$. Then the segmented results with gaps represented as follows:

$$C_{1:e_1}|G_{1:e_1-1}, G_{e_1}, C_{e_1+1:e_2}|G_{e_1+1:e_2-1}, G_{e_2}, \cdots, G_{e_{m-1}}, C_{e_{m-1}+1:e_m}|G_{e_{m-1}+1:e_m-1}$$

[1] Jiang et al. (2009) describe a similar situation. We also tried to directly adapt a classifier trained with news corpus to Chinese medicine patent corpus, which led to dramatic decrease in accuracy.

where $C_{i:j}|G_{i:j-1}$ denotes the character-gap sequence with gaps $G_{i:j-1}$. As is shown above, there are two kinds of gaps, one occurs inside a word, such as $G_1 \cdots G_{e_1-1}$; the other occurs between two words, such as G_{e_1}. Word Segmentation can be treated as a gap tagging problem.

2.1 From Character Tagging to Gap Tagging

Ng and Low (2004) give a boundary tag to each character denoting its relative position in a word. There are four boundary tags: "b" for a character that begins a word, "m" for a character that occurs in the middle of a word, "e" for a character that ends a word, and "s" for a character that occurs as a single-character word. Following Ng and Low (2004), the feature templates and the corresponding instances are listed in Table 1.

Feature Template	**Instances**
$C_i (i = -2, \cdots, 2)$	$C_{-2} =$ 美, $C_{-1} =$ 国, $C_0 =$ 商, $C_1 =$ 务, $C_2 =$ 部
$C_i C_{i+1} (i = -2, \cdots, 1)$	$C_{-2}C_{-1} =$ 美国, $C_{-1}C_0 =$ 国商, $C_0C_1 =$ 商务, $C_1C_2 =$ 务部
$C_{-1}C_1$	$C_{-1}C_1 =$ 国务
$Pu(C_0)$	$Pu(C_0) = 0$
$T(C_{-2})T(C_{-1})T(C_0)T(C_1)T(C_2)$	$T(C_{-2})T(C_{-1})T(C_0)T(C_1)T(C_2) = 44444$

Table 1: Character tagging feature templates and the corresponding instances, suppose we are considering the third character "商" in "美国商务部".

Actually, the classical character tagging method can be explained by gap tagging. Given a character-tag sequence $\cdots G_{i-1}C_iG_i \cdots$, tagging $G_{i-1}G_i$ to "AA" is equivalent to assigning C_i to "m", "AS" is equal to "e", "SA" is equal to "b" and "SS" is equal to "s".

2.2 N-Gram Markov Gap Tagging

For n-gram markov gap tagging, the number of decisions is 2^{N+1} when we consider N+1 gaps in one step. These decisions compose a decision set, denoted as $\{A,S\}^{\bigotimes N+1}$. The operator $\bigotimes$ stands for the cartesian product between different decision sets. Gap sequence $G_{i-N:i}$ will be tagged in the position G_i with n-gram markov gap classification. Given a character-gap sequence: $x = C_{1:n}|G_{1:n-1}$, we aim to find an output $F(x)$ that satisfies:

$$F(x) = \underset{y \in \{A,S\}^{\bigotimes n-1}}{\arg\max}\ Score(x, y) \tag{1}$$

where $Score$ stands for the score of the tagging sequence y evaluated by the classifier.

$$Score(x, y) = \sum_i Eval(y_{i-N:i}, \Phi(x, i)) \tag{2}$$

$\Phi(x, i)$ denotes the feature vector extracted from position G_i of character-gap sequence x. $Eval$ denotes the evaluated score to $y_{i-N:i}$ selected by classifier, based on feature vector $\Phi(x, i)$.

We use the Maximum Entropy approach to train the classifier, with dynamic programming decoding algorithm to find the highest score. In our experiments, we use 2-gram markov gap tagging, with trading off the performance and speed.

Gap tagging features are denoted by the characters around the gap. When we consider the current gap G_0, the ith character to the left of G_0 is C_{1-i}, and to the right is C_i. The gap tagging feature templates and the corresponding instances are listed in Table 2.

Feature Template	Instances
$C_i (i = 0, \cdots, 1)$	$C_0 =$ 商, $C_1 =$ 务
$C_i C_{i+1} (i = -1, \cdots, 1)$	$C_{-1}C_0 =$ 国商, $C_0C_1 =$ 商务, $C_1C_2 =$ 务部

Table 2: Gap tagging feature templates and the corresponding instances, suppose we are considering the gap between the character " 商" and the character "务" in "美国商务部".

3 Dictionary-Based Feature Extraction

With annotated corpora, discriminative training approaches for CWS have two stages. Extracting feature instances and training a model with them. Since the training algorithm has been described in the last section, now we describe the method of dictionary matching based feature extraction by a dictionary and a raw corpus. This method contains two key points, the dictionary matching strategy (Section 3.1) and the tagging strategy (character tagging or gap tagging) of extracting feature instances (Section 3.2).

3.1 Dictionary Bi-direction Maximum Matching

Noting that the dictionary we used does not contain time words, numerals and English strings, which can be accurately recognized by some manual rules. Besides, single-character words are not included in the dictionary for two reasons: First, the single-character word is very flexible and usually appears as a character in a multi-character word. Second, the scale of dictionary is limited and single-character words are in high frequence, a lot of matching errors will occur during dictionary matching. [2]

To minimize errors caused by the dictionary matching ambiguity, we use forward maximum matching (FMM) and backward maximum matching (BMM) to match the raw corpus with the dictionary, to generate feature instances sets S_{fmm} and S_{bmm}, respectively. Then, we reserves the intersection of the two feature instances sets to the reliable set $S_{fmm\&bmm}$.

The dictionary matching procedure is that, we scan each sentence S in the raw corpus R, suppose $S = C_{1:N}(C_q(q = 1, \cdots, N)$ is character), and compare the continuous characters $C_{i:k}(0 \leq i < k \leq N)$ with the words in dictionary D by FMM and BMM, respectively. If successfully matched, it means that the character sequence $C_{i:k}$ is matched with the word W_j ($W_j = C_{i:k}$) in dictionary. Then we transform the character sequence $C_{i:k}$ and its context (*preceding-text* and *following-text* are both made up of two characters, $C_{i-2}C_{i-1}$ is the *preceding-text*, $C_{k+1}C_{k+2}$ is the *following-text*) to a *text fragment P* with the structure of "*preceding-text*($C_{i-2}C_{i-1}$)+*word*($C_{i:k}$)+*following-text*($C_{k+1}C_{k+2}$)".

3.2 Gap Tagging Feature Extraction

3.2.1 Features from Multi-Character Words

It is worth mentioning that the feature instances extracted by the method described in Section 3.1 are from multi-character words. For each *text fragment* "*P=preceding-text*($C_{i-2}C_{i-1}$)+*word*($C_{i:k}$)+*following-text*($C_{k+1}C_{k+2}$)" extracted by FMM and BMM, we extract feature instances from the gap in three kinds:

1. Gap between the last character of the *preceding-text* and the first character of the *word*

[2] We test the performance of our method, with the dictionary containing single-character words. As single-character words lead to errors during dictionary matching, the performance is poorer than DMM method.

2. Gap between every pair of characters in the *word*
3. Gap between the last character of the *word* and the first character of the *following-text*

DMM Strategy	FMM	BMM
Result	美国 商务部 长 访问 上海	美国 商务 部长 访问 上海

Table 3: Results of sentence "美国商务部长访问上海" by FMM and BMM, suppose words "美国","商务部","商务","部长","访问","上海" are included in the dictionary.

Method	Fragments	Feature Instance
FMM	美国 商务部 长访	$S: C_0 = 国, C_1 = 商, C_{-1}C_0 = 美国, C_0C_1 = 国商, C_1C_2 = 商务$
		$A: C_0 = 商, C_1 = 务, C_{-1}C_0 = 国商, C_0C_1 = 商务, C_1C_2 = 务部$
		$A: C_0 = 务, C_1 = 部, C_{-1}C_0 = 商务, C_0C_1 = 务部, C_1C_2 = 部长$
		$S: C_0 = 部, C_1 = 长, C_{-1}C_0 = 务部, C_0C_1 = 部长, C_1C_2 = 长访$
BMM	美国 商务 部长	$S: C_0 = 国, C_1 = 商, C_{-1}C_0 = 美国, C_0C_1 = 国商, C_1C_2 = 商务$
		$A: C_0 = 商, C_1 = 务, C_{-1}C_0 = 国商, C_0C_1 = 商务, C_1C_2 = 务部$
		$S: C_0 = 务, C_1 = 部, C_{-1}C_0 = 商务, C_0C_1 = 务部, C_1C_2 = 部长$
	商务 部长 访问	$S: C_0 = 务, C_1 = 部, C_{-1}C_0 = 商务, C_0C_1 = 务部, C_1C_2 = 部长$
		$A: C_0 = 部, C_1 = 长, C_{-1}C_0 = 务部, C_0C_1 = 部长, C_1C_2 = 长访$
		$S: C_0 = 长, C_1 = 访, C_{-1}C_0 = 部长, C_0C_1 = 长访, C_1C_2 = 访问$
FMM &BMM	—	$S: C_0 = 国, C_1 = 商, C_{-1}C_0 = 美国, C_0C_1 = 国商, C_1C_2 = 商务$
		$A: C_0 = 商, C_1 = 务, C_{-1}C_0 = 国商, C_0C_1 = 商务, C_1C_2 = 务部$

Table 4: Gap tagging feature instances sets extracted by FMM, BMM and the intersection of the two sets, for two matching results of character sequence "商务部长" listed in Table 3. "S" means "split", and "A" means "adjoin". "—" means the corresponding "Fragments" are undefined.

For example, Table 3 lists the segment results of the sentence "美国商务部长访问上海", by FMM and BMM. It is difficult to distinguish which one is better. So we extract feature instances from both the matching results, and take the intersection of the two feature instances sets. For the two results listed in Table 3, only four characters' ("商", " 务", "部", " 长") segmentation results are different. We only list the feature instances sets of "商务部长" extracted by FMM and BMM and the intersection (corresponding to "FMM&BMM") of the two sets in Table 4. As the segmented results of "商务部长" generated by BMM are " 商务" and " 部长", two fragments are constructed. Besides, features of "长" (single-character word) are not extracted by FMM.

Since we can generate feature instances from each character in the matching word ($C_{i:k}$), we can also extract character tagging feature instances from the *text fragment*.

3.2.2 Features from Single-Character Words

As we have described in section 3.1, single-character words are not included in the dictionary. Lacking feature instances from single-character words, the ability of character tagging classifier weakened a lot, since the feature instances with tagger "s" can not be extracted. Gap tagging does not need to distinguish whether a character is a single-character word, so we do not have to extract feature instances of single-character words.

However, single-character words account for a large proportion in actual corpora. The ability of classifier will be stronger if adding feature instances of single-character words. We propose a simple strategy to extract feature instances of single-character words. Given a Character "*C*", if its left part and right part are words in dictionary, or time words, or numerals, or English

strings, or the beginning tag of the sentence, or the ending tag of the sentence, then we take "C" as a single-character word. Next, we can extract feature instances from single-character words as described in section 3.1.

4 Experiments

4.1 Setup

We conducted experiments mainly on two domains:

1. In news domain, we use Penn Chinese Treebank 5.0 (CTB). The dictionary D_{ctb} (33500 words) consists of words extracted from chapters 1-270 (18K sentences). The raw corpus[3] is mainly in news domain with 10M sentences. In order to compare with CMP, we also extract a smaller scale raw text with 2M sentences from the raw corpus (10M).
2. In Chinese medicine patent (CMP) domain, the dictionary D_{cmp} (21800 words) is from our internal resource, consists of words in Chinese medicine patent domain. As the dictionary D_{cmp} is so specialized with few common words, we combine D_{cmp} and D_{ctb} to a new dictionary $D_{cmp+ctb}$ (55100 words after duplicate removal). The raw corpus is from our internal resource. We extract 300 sentences from the raw corpus, and annotate them manually to form the testing set, the others (2M sentences) are used for training.

Dictionaries in both domains do not contain time words, numerals, English strings and single-character words.

We use the Maximum Entropy Toolkit developed by Zhang[4] to train the discriminative model. Empirically, the number of training iterations is 150 with gaussian prior 3.0 on training process. The performance measurement indicators for word segmentation is balanced *F-measure*, $F = 2PR/(P+R)$, a function of *Precision* P and *Recall* R. *Precision* is the relative amount of correct words in the system output. *Recall* is the relative amount of correct words compared to the gold standard annotations.

4.2 Dictionary Based Discriminative Training Results

Given a sentence, we first recognize time words, numerals and English strings by manual rules, then process other parts by the discriminative classifier.

Train On		**Test On (F_1%)**	
Dictionary	S-W	Character-tagging	Gap-tagging
D_{cmp}	No	56.21	67.55
D_{cmp}	Yes	56.21	67.51
$D_{cmp+ctb}$	No	53.13	76.23
$D_{cmp+ctb}$	Yes	55.47	76.37

Table 5: Results on CMP by dictionary matching based discriminative classifier with gap tagging and character tagging. "S-W" means feature instances of single-character words, "Yes" stands for containing these feature instances, "No" for not.

First, we check which tagging strategy is more suitable for our method, gap-tagging or character-tagging? In Table 5, it is clear from each row that results generated by gap-tagging

[3]The raw corpus includes People's Daily corpus(removing spaces); LDC2002E18, LDC2003E07, LDC2003E14; Hansards portion of LDC2004T07, LDC2004T08 , LDC2005T06; and other raw corpora of our internal resource.

[4]homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html

classifier achieve much higher F_1 scores than those generated by character-tagging, with +16.66 points on average. Although, adding features instances of single-character words does improve the performance of both gap-tagging (from 76.23 to 76.37) and character-tagging (from 53.13 to 55.47)[5]. But the accuracy increment brought by this strategy can not compensate for the defects of character-tagging. Thereby, gap-tagging classification is more suitable for our method.

Method		**Test On (F_1%)**	
		CTB	CMP
FMM		91.05	72.30
BMM		91.44	72.63
Dictionary	**Raw corpus / scale**		
D_{ctb}	News / 2M	92.98	—
D_{ctb}	News / 10M	**93.28**	—
D_{cmp}	CMP / 2M	—	67.51
$D_{cmp+ctb}$	CMP / 2M	—	**76.37**

Table 6: Comparisons between DMM and our method (including feature instances of single-character words).

Then, we propose the comparisons between DMM and our method. In Table 6, when we use FMM or BMM on CTB testing set, the corresponding dictionary is D_{ctb}, and $D_{cmp+ctb}$ for CMP testing set. Comparing row 1 and row 2, we can see that BMM achieves better performance than FMM in both CTB and CMP. On CTB test, our method achieves an increment of 1.54 points on F_1 score than BMM (from 91.44 to 92.98), with small scale raw corpus. Using the larger scale raw corpus (10M sentences) leads to a further increment of 0.3 points (93.28), with error reduction[6] of 21.50%. The result of row 5 (67.51) is lower than row 1 (72.30), the reason is that D_{cmp} contains few common words, so that few feature instances of common words can be extracted. When using $D_{cmp+ctb}$, we get higher performance (76.37).

Dictionary	**Raw corpus / scale**	**Test On (Recall %)**	
		CTB	CMP
D_{ctb}	News / 10M	**42.54**	—
$D_{cmp+ctb}$	CMP / 2M	—	**23.72**

Table 7: Recall rate of OOV words by our method.

To compare with DMM on the ability of OOV words recognition, we show the recall rate of OOV words by our method in Table 7. As time words, numerals and English strings can be recognized by manual rules, we do not consider them when computing recall rate of OOV words. Besides, we do not consider single-character words which are not included in the dictionary. For DMM method, the recall rates of OOV words are zero both in CTB and CMP. Compared with DMM, our method shows much stronger ability on OOV words recognition, with the recall rate increments of 42.54% and 23.72%, respectively in CTB and CMP.

Due to the obvious improvement brought by our method, we can safely conclude that our dictionary matching based discriminative training approach is better than DMM method. With no available annotated corpora, our method achieves considerable performance.

[5]Results of row 1 and row 2 are not consistent with this situation, as the dictionary D_{cmp} contains much more specialized vocabulary than common words, resulting in much fewer feature instances of common words.

[6]Error rate is defined as $1 - F_1$, which has been described in many previous works.

5 Related Work

Many works have been devoted to the word segmentation task in recent years, including the word-based perceptron algorithm (Zhang and Clark, 2007); taking punctuation as implicit annotations (Li and Sun, 2009); the strategies of stacked modeling (Sun, 2011); the investigation of word structures (Li, 2011); the approach of automatic adaptation between different corpora (Jiang et al., 2009, 2012); joint model on word segmentation and new word detection (Sun et al., 2012) and other single-model approach (Zhang and Clark, 2008, 2010; Kruengkrai et al., 2009; Wang et al., 2010).

There are also some unsupervised approaches with raw text. Peng and Schuurmans (2001) propose an unsupervised approach based on an improved expectation maximum learning algorithm and a pruning algorithm based on mutual information. Non-parametric Bayesian techniques (Johnson and Goldwater, 2009; Mochihashi et al., 2009) have been introduced to word segmentation. Bootstrapped voting experts algorithm paired with minimum description length is used to for word segmentation (Hewlett and Cohen, 2011). Wang et al. (2011) propose an ESA (Evaluation, Selection, and Adjustment) unsupervised approach to word segmentation.

Our method is different from above methods, as we integrate the advantages of discriminative training and DMM. Moreover, we only use a dictionary[7] and a raw text.

Conclusion

In this paper, we propose an effective approach of integrating the advantages of discriminative training and DMM, to build a high quality word segmenter with only a dictionary and a raw text. We conduct experiments in CWS on both news domain and Chinese medicine patent domain. Our method gains much higher word segmentation accuracy than DMM in both domains, with error reductions of 21.50% and 13.66%, respectively. The capability of OOV words recognition of our method is stronger than DMM by a large margin, with the increments of recall rate 42.54% and 23.72%, respectively.

Our method does not use annotated corpora, we only use a small scale dictionary and a raw text, which are easier to get in resource-poor languages and domains compared with annotated corpora. Theoretically, our method is not only effective in Chinese, but also in languages with no obvious word delimiters in sentences, such as Japanese and some other Asian languages.

In the future, we will explore better strategies of extracting high quality features from raw text. Besides, we will try to integrate some unsupervised approaches to our method, which may help us learn more knowledge from raw text.

Acknowledgments

The authors were supported by National Natural Science Foundation of China, Contracts 61202216, and 863 State Key Project No. 2011AA01A207. We are grateful to the anonymous reviewers for their thorough reviewing and informative comments.

References

Collins, M. (2002). Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical*

[7] The dictionaries used in this paper are available resources. To our knowledge, we can produce a dictionary for a novel domain/language from Wikipedia or Baidupedia (Chinese).

methods in natural language processing-Volume 10, pages 1–8. Association for Computational Linguistics.

Collins, M. and Roark, B. (2004). Incremental parsing with the perceptron algorithm. In *Proceedings of ACL*, volume 2004.

Hewlett, D. and Cohen, P. (2011). Fully unsupervised word segmentation with bve and mdl. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 540–545. Association for Computational Linguistics.

Jiang, W., Huang, L., and Liu, Q. (2009). Automatic adaptation of annotation standards: Chinese word segmentation and pos tagging: a case study. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 522–530. Association for Computational Linguistics.

Jiang, W., Meng, F., Liu, Q., and Lü, Y. (2012). Iterative annotation transformation with predict-self reestimation for chinese word segmentation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 412–420.

Johnson, M. and Goldwater, S. (2009). Improving nonparameteric bayesian inference: experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 317–325. Association for Computational Linguistics.

Kruengkrai, C., Uchimoto, K., Kazama, J., Wang, Y., Torisawa, K., and Isahara, H. (2009). An error-driven word-character hybrid model for joint chinese word segmentation and pos tagging. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 513–521. Association for Computational Linguistics.

Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th ICML*, pages 282–289.

Li, Z. (2011). Parsing the internal structure of words: a new paradigm for chinese word segmentation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. Portland, Oregon, USA: Association for Computational Linguistics*.

Li, Z. and Sun, M. (2009). Punctuation as implicit annotations for chinese word segmentation. *Computational Linguistics*, 35(4):505–512.

Mochihashi, D., Yamada, T., and Ueda, N. (2009). Bayesian unsupervised word segmentation with nested pitman-yor language modeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 100–108. Association for Computational Linguistics.

Ng, H. and Low, J. (2004). Chinese part-of-speech tagging: One-at-a-time or all-at-once? word-based or character-based. In *Proceedings of EMNLP*, volume 4.

Peng, F. and Schuurmans, D. (2001). Self-supervised chinese word segmentation. *Advances in Intelligent Data Analysis*, pages 238–247.

Ratnaparkhi and Adwait (1996). A maximum entropy model for part-of-speech tagging. In *Proceedings of the conference on empirical methods in natural language processing*, volume 1, pages 133–142.

Sun, W. (2011). A stacked sub-word model for joint chinese word segmentation and part-of-speech tagging. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 1385–1394.

Sun, X., Wang, H., and Li, W. (2012). Fast online training with frequency-adaptive learning rates for chinese word segmentation and new word detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 253–262.

Wang, H., Zhu, J., Tang, S., and Fan, X. (2011). A new unsupervised approach to word segmentation. *Computational Linguistics*, 37(3):421–454.

Wang, K., Zong, C., and Su, K. (2010). A character-based joint model for chinese word segmentation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1173–1181. Association for Computational Linguistics.

Xue, N. and Shen, L. (2003). Chinese word segmentation as lmr tagging. In *Proceedings of the second SIGHAN workshop on Chinese language processing-Volume 17*, pages 176–179. Association for Computational Linguistics.

Xue, N., Xia, F., Chiou, F., and Palmer, M. (2005). The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(02):207–238.

Zhang, Y. and Clark, S. (2007). Chinese segmentation with a word-based perceptron algorithm. In *Anual Meeting-Association for Computational Linguistics*, volume 45, page 840.

Zhang, Y. and Clark, S. (2008). Joint word segmentation and pos tagging using a single perceptron. In *Proceedings of ACL*, volume 8, pages 888–896.

Zhang, Y. and Clark, S. (2010). A fast decoder for joint word segmentation and pos-tagging using a single discriminative model. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 843–852. Association for Computational Linguistics.