Combining Machine Learning with Linguistic Heuristics for Chinese Word Segmentation

Xiaofei Lu

Department of Linguistics and Applied Language Studies
The Pennsylvania State University
University Park, PA 16802, USA
xxl13@psu.edu

Abstract

This paper describes a hybrid model that combines machine learning with linguistic heuristics for integrating unknown word identification with Chinese word segmentation. The model consists of two components: a position-of-character (POC) tagging component that annotates each character in a sentence with a POC tag that indicates its position in a word, and a merging component that transforms a POC-tagged character sequence into a word-segmented sentence. The tagging component uses a support vector machine based tagger to produce an initial tagging of the text and a transformation-based tagger to improve the initial tagging. In addition to the POC tags assigned to the characters, the merging component incorporates a number of linguistic and statistical heuristics to detect words with regular internal structures, recognize long words, and filter non-words. Experiments show that, without resorting to a separate unknown word identification mechanism, the model achieves an F-score of 95.0% for word segmentation and a competitive recall of 74.8% for unknown word recognition.

Introduction

Word segmentation is the initial step of almost any text analysis task. In languages where word boundaries are marked by whitespace and punctuation marks, word segmentation is relatively straightforward. However, there are no unambiguous word boundary markers in Chinese, and Chinese word segmentation is a nontrivial task. The task is further complicated by the lack of a commonly accepted definition of word in Chinese among theoretical linguistics. The Chinese language processing community generally adopts a rather pragmatic approach to this issue, where the definition of word varies depending on the purpose of the natural language processing system and the segmentation standard it adopts (Sproat et al. 1996).

Chinese word segmentation involves two issues: segmentation ambiguity resolution and unknown word identification. Most previous studies treat these as two separate problems, using a mechanism to identify unknown words in a post-processing step after word segmentation is done. However, determining where word boundaries are necessarily involves understanding how characters relate to and interact with each other in context, and it is desirable to capture this dynamic interaction by integrating unknown word identification with word segmentation. Several recent studies have taken a unified approach to unknown word identification and word segmentation (e.g., Sproat et al. 1996; Xue 2003; Gao et al. 2005).

We describe a hybrid model that combines machine learning with linguistic heuristics for integrating unknown word identification with Chinese word segmentation. We adopt the notion of character-based tagging (Xue 2003) to directly model the combinatory power of Chinese characters, i.e., the tendency for characters to combine with adjacent characters to form words, either known or unknown, in different contexts. The model consists of two components. First, a position-of-character (POC) tagging component tags each character in a sentence with a POC tag that indicates its position in a word. This component uses a support vector machine based tagger to produce an initial tagging of the text and a transformation-based tagger to improve the initial tagging. Second, a merging component transforms a POC-tagged character sequence into a word-segmented sentence, using a number of linguistic and statistical heuristics to detect words with regular internal structures, recognize long words, and filter non-words. Without resorting to a sophisticated mechanism for unknown word identification or additional resources other than a word-segmented training corpus, the model achieves an F-score of 95.0% for word segmentation and a competitive recall of 74.8% for unknown word identification.

The rest of the paper is organized as follows. Section 2 reviews previous approaches to Chinese word segmentation. Section 3 details the two components of the proposed model. Section 4 discusses the experiment results of the model. Section 5 concludes the paper and points to avenues for future research.

Previous Studies

Previous approaches for word segmentation fall into four categories: dictionary-based, statistical, statistical and dictionary-based, and machine learning approaches. In dictionary-based approaches, only words listed in the dictionary are identified. Most studies from these approaches use some variation of the maximum matching algorithm along with heuristics to handle segmentation ambiguities (e.g., Nie et al. 1995). These approaches require a separate mechanism for unknown word identification and their performance depends heavily on the quality of the dictionary. Statistical approaches use information-theoretical or probabilistic measures to determine whether adjacent characters form words or which segmentation is most likely for a sentence (e.g., Ge et al. 1999). These approaches incorporate little linguistic knowledge and generally perform worse than other approaches. Statistical and dictionary-based approaches attempt to benefit from both worlds, using both the information about words in the dictionary and statistical information derived from the corpus to compute the most likely segmentation of a sentence. However, it is generally agreed that the quality of the base lexicon is more important than the model and that unknown words constitute the greatest challenge (e.g., Sproat et al. 1996). With the availability of word-segmented training corpora, a number of supervised machine learning algorithms have been applied to Chinese word segmentation, including, e.g., maximum entropy (Xue 2003), conditional random fields (e.g., Tseng et al. 2005), and linear mixture models (Gao et al. 2005). These approaches are able to integrate unknown word identification with word segmentation and have achieved fairly competitive results.

Proposed Approach

This section describes a model that combines machine learning with linguistic heuristics to integrate unknown word identification with word segmentation. The main hypothesis tested here is that the notion of character-based tagging (Xue 2003) can be used to directly model the combinatory power of Chinese characters to combine with adjacent characters to form words in different contexts and to integrate unknown word identification with word segmentation. The model consists of two components. First, a position-of-character (POC) tagging component tags each character in a sentence with a POC tag that indicates its position in a word. This component is based on the transformation-based learning (TBL) algorithm (Brill 1995), using a tagger based on support vector machines (SVMs) (Vapnik 1995) as an initial tagger for the algorithm. Second, a merging component transforms a POC-tagged character sequence into a word-segmented sentence, using a number of linguistic and statistical heuristics to handle several special types of words.

The Tagging Component

The tagset defined for the tagging component consists of four tags: *L*, *M*, *R*, and *W*, each of which indicates that the character is in a word-initial, word-middle, or word-final position or is a monosyllabic word.

A Transformation-Based Learning Tagger. The TBL algorithm is adopted for the tagging component because, compared with other statistical machine learning algorithms, it captures linguistic knowledge in a more direct fashion without compromising performance (Brill 1995). The implementation of the algorithm used in this study is fnTBL (Ngai and Florian 2001), which is more efficient than Brill's original implementation.

The TBL algorithm requires a tagged training corpus (the *truth*) and its corresponding raw version. A tagged corpus can be converted from a word-segmented corpus by assigning each character a tag based on its position in the word containing it. The conversion process is illustrated by the following example, where the word-segmented sentence in (1a) is converted into a tagged character sequence in (1b).

(1) a. 今天 是 星期一 .
Today is Monday .
'Today is Monday.'

b. 今/L 天/R 是/W 星/L 期/W 一/R ./W

In addition to the tagged training corpus and the corresponding raw corpus, the algorithm requires three components. The first is an initial tagging of the raw corpus. Although the algorithm places no requirement on the initial tagger, previous studies have shown that a better initial tagger leads to better final results and shorter learning time (e.g., Hockenmaier and Brew 1998). For this reason, we use a sophisticated SVM-based initial tagger. A second initial tagger based on the hidden Markov model (HMM) is used for comparison.

The second component is the space of transformations allowed. Each transformation consists of a rewrite rule and a triggering environment. The set of transformations used in this study is similar to the set Ngai and Florian (2001) defined for the task of base NP chunking. In this case, however, the triggering context is defined over characters and POC tags. The triggering context considered include the character and tag in the current position and those in the three positions immediately preceding or following the current position. Three transformations are given in (2) as an illustration.

(2) Change the current tag t_i to t_j, if the current tag is x, the current character is a, and one of the following is true:
   a. the preceding (following) character is b
   b. the tag two positions to the left (right) is z
   c. the previous (following) character is b, and the previous (following) tag is y

The third component is a scoring function, which is used to compare the corpus to the *truth* and determine which transformation should be learned. The function we use is the number of tagging error reductions achieved after applying a transformation.

Once all the components are in place, the iterative training process takes place as follows. At each iteration, the learner applies each possible instantiation of the transformation templates to the text (starting with the text tagged by the initial tagger), counts the number of tagging error reductions each transformation achieves, and chooses the transformation that achieves the greatest number of tagging error reductions. That transformation is applied to the text, and the learning process repeats until no more transformations reduce errors beyond a pre-determined threshold. The output of the algorithm is a ranked list of transformations that can be applied to new text.

An SVM-Based Initial Tagger. SVMs are binary classifiers on a feature vector space R^L. Given a set of training data, $\{(x_i, y_i) | x_i \in R^L, y_i \in \{\pm 1\}, 1 \le i \le l\}$, where x_i is the ith sample in the training data and y_i is the label of x_i, a hyperplane, given in (3), separates the set into two classes in such a way that the constraints in (4) are satisfied:

$$(3) \quad w \cdot x + b = 0, w \in R^L, b \in R$$

$$(4) \quad y_i \cdot (w \cdot x_i + b) \ge 1$$

In (3) and (4), w is a weight vector with one weight for each feature, and b is a bias, which is the distance of the hyperplane to the origin. Among all hyperplanes that separate the training data into two sets, SVMs find the optimal hyperplane with maximal margin, i.e., maximal distance between the hyperplane and the nearest positive and negative samples, because it is expected to minimize expected test error. Giving a test example x, its label y is determined by the sign of a discrimination function $f(x)$ given by the SVMs classifier as follows:

$$(5) \quad f(x) = \mathrm{sgn}\Big(\sum_{z_i \in SV} \alpha_i y_i K(x, z_i) + b\Big)$$

where $b \in R$, z_i is a support vector, which receives a non-zero weight α_i, $K(x, z_i)$ is a polynomial kernel function of degree d given by $K(x, z_i) = (x \cdot z_i + 1)^d$, which maps vectors into a higher dimensional space where all combinations of up to d features are considered, and SV denotes the set of support vectors, i.e., the vectors that receive a non-zero weight. The support vectors and the parameters are determined by quadratic programming. If $f(x) = +1$, then x is a positive member, otherwise it is a negative member.

The implementation of the SVM classifier used is SVMTool (Giménez and Màrquez 2004). As SVMs are binary classifiers, some adaptation is necessary to make them suitable for multi-class classification tasks. Giménez and Màrquez used a one-per-class binarization, where they trained an SVM for each class to determine whether an example is of this class or not. At classification time, the most confident tag given by all SVMs is selected.

The features used for tagging include character and tag n-grams. The two characters and tags preceding and following the current character and tag are considered. At running time, however, the tags of the characters to the right of the current character are not known. In SVMTool, a general ambiguity-class tag, i.e., a label that concatenates all the possible tags for a character, is used for the right context characters. Table 1 summarizes the feature set used for tagging. Only the POC features are used for unknown characters.

Character features	c_{-2}, c_{-1}, c_0, c_1, c_2
POC features	t_{-2}, t_{-1}
Ambiguity class	a_0, a_1, a_2
Character bigrams	(c_{-2},c_{-1}), (c_{-1},c_1), (c_{-1},c_0), (c_0,c_1), (c_1,c_2)
POC bigrams	(t_{-2},t_{-1}), (t_{-1},a_1), (a_1,a_2)
Character trigrams	(c_{-2},c_{-1},c_0), (c_{-2},c_{-1},c_1), (c_{-1},c_0,c_1), (c_0,c_1,c_2), (c_{-1},c_1,c_2)
POC trigrams	(t_{-2},t_{-1},a_0), (t_{-2},t_{-1},a_1), (t_{-1},a_0,a_1), (t_{-1},a_1,a_2)

Table 1: Feature set for the SVM-based tagger

An HMM-Based Initial Tagger. To compare the impact of the initial tagger on the performance of the TBL algorithm, a first-order HMM tagger (Charniak et al. 1993) is implemented. This model computes the most likely tag sequence $t_1 \ldots t_n$ for a character sequence $c_1 \ldots c_n$, as follows:

$$(6) \quad \arg\max_{t_i \ldots t_n} \prod_{i=1}^{n} p(t_i \mid t_{i-1}) p(t_i \mid c_i)$$

where t_i and c_i denote the ith tag in the tag sequence and the ith character in the character sequence respectively, $p(t_i|t_{i-1})$ denotes the transition probability, i.e., the probability of a tag given its previous tag, and $p(t_i|c_i)$ denotes the lexical probability, i.e., the probability of a tag given a character. The transition and lexical probabilities are estimated from the training corpus. For unknown characters, the lexical probabilities are uniformly distributed among the four tags defined in the tagset. The transition probabilities are not smoothed, as all unseen tag bigrams, such as (R,R) and (W,R), are impossible combinations. Once the model is trained, the Viterbi algorithm (Rabiner 1989) is used to tag new text.

The Merging Component

The merging component transforms a tagged character sequence into a word-segmented sentence. By default, it inserts a word boundary marker after each character tagged R (word-final) or W (monosyllabic word). It incorporates several linguistic and statistical heuristics for 1) detecting three types of unknown words with regular internal structures, i.e., numeric type compounds, reduplicated and

derived words, and transliterated foreign names; 2) recognizing long words that have occurred in the training corpus; and 3) filtering non-words.

Numeric Type Compounds. The component uses regular expressions to detect numeric type compounds such as dates, times, fractions, numbers, etc. The patterns for different types of numeric type compounds are generalized from the training corpus, and a character string that fits one of these patterns is grouped into one segmentation unit. Gao et al. (2005) showed that the performance of detecting such words using their internal properties is comparable with that of using contextual information.

Reduplicated and Derived Words. The component also uses heuristics to detect reduplicated words that are three or more characters long and some derived words with predictable internal structures. If a character string fits a reduplication pattern in Chinese, it is grouped as one word. For derived words, many morphemes are ambiguous between an affix and a word, e.g., 家 can be either an affix '-ist' or a noun 'family'. Detection of the correct use of such morphemes in bisyllabic words is hard. To avoid overgeneration, the heuristics only detect derived words formed with a multisyllabic root word and an unambiguous affix, such as 学家 study-expert '-ist'. If an unambiguous prefix or suffix is detected, it is attached to the following or previous word, if that word is at least two characters long.

Transliterated Foreign Names. Foreign names are usually transliterated using a subset of Chinese characters. Most transliterated names consist of three or more characters, and these words pose a challenge to the tagger. We acquire a list of characters used for transliterations from the wordlist generated from the training corpus. There is a dot used exclusively within transliterated foreign names to indicate different parts of a name. For example, *George Bush* is transliterated as 乔治·布什, where the dot within the name indicates that 乔治 'George' and 布什 'Bush' are different parts of the same name. Based on this simple observation, the algorithm proceeds as follows:

1) Extract names with the dot from the wordlist and store all characters in these names in a seed list.
2) Extract all candidate names that are four or more characters long with all but one character from the seed list. For each candidate, add the one character not on the seed list to the seed list. Repeat until no more characters can be acquired.
3) Filter out characters acquired in step 2 that have appeared only in one candidate.

In processing new text, the algorithm first uses the final character list to identify candidate names and then filters candidates using contextual information, as follows:

1) Add characters immediately before or after the dot in the text to the final list of characters, if they are not already included.
2) Identify character *n*-grams ($n \geq 3$) whose component characters are all from the list.
3) Extract five characters to its left and right.
4) Strip the leftmost character of the *n*-gram if it forms a word with the one, two, three, four, or five left context characters and add it to the left context. Iterate until the leftmost character of the *n*-gram no longer forms a word with its left context characters.
5) Strip the leftmost two characters of the n-gram if they form a word with the one, two, three, four, or five left context characters and add them to the left context. Iterate until the leftmost two characters of the *n*-gram no longer form a word with its left context characters.
6) Do steps 4 and 5 with the rightmost characters of the *n*-gram and the right context characters.
7) If the final *n*-gram has three or more characters, it is considered a transliterated foreign name.

Long Words. Another hypothesis tested in this research is that longer words behave more consistently than shorter words. In particular, we hypothesize that if a string of four or more characters appeared in the training corpus as a word, it is likely to be a word in the test corpus. Whereas there are foreseeable counterexamples for this hypothesis, its usefulness will be empirically tested. Given the fuzzy line between compounds and phrases in Chinese, a character string may be considered as a compound in one segmentation standard but a phrase in another. For example, the string 个人所得税 'personal income tax' may be considered one word or three words, 个人 'personal', 所得 'income', and 税 'tax', depending on the standard. Within the same standard, however, such strings should be treated consistently. To adapt to the standard following which the corpus is segmented, we use the wordlist generated from the training corpus instead of an existing lexicon. If a string of four or more characters is found in the wordlist, the merging component considers it a word.

Non-Word Filtering. The last type of heuristics is for non-word filtering. The first of these is used to detect bisyllabic non-words. If the tagger tags two adjacent characters as *L* (word-initial) and *R* (word-final), the string is a candidate new word, if it has not occurred in the training data. The candidate is filtered in two steps. First, it is checked against two lists of characters that contain frequent, unproductive morphemes that do not occur in 1) word-initial position of new bisyllabic words, e.g., 而 'but', or 2) word-final position of new bisyllabic words, e.g., 这 'this', respectively. If the first or second character of the candidate is on the first or second list, respectively, the candidate is split into two words. Second, the probability for a character to appear in word-initial or word-final position is estimated from the training data as in (7):

$$P(C, Pos) = \frac{F(C, Pos)}{F(C)} \quad (7)$$

where C denotes a character, Pos denotes a position in a word, $P(C,Pos)$ is the probability that C occurs in Pos, $F(C,Pos)$ is the number of times C occurs in Pos, and $F(C)$ is the number of times C appears in any position of a word. Given a candidate new word, the probability for it to be a word is computed as the joint probability of $P(C,Pos)$ for both of its component characters. If the joint probability is below a pre-determined threshold, then the candidate is considered a non-word and is split into two words.

Finally, the following two rules are used to merge or split character strings of certain patterns, based on the error analysis in the development stage. First, if two adjacent characters are both tagged *W* (monosyllabic word), but they have always occurred in the training data as a word, they are merged into a word. Second, if three adjacent characters are tagged *W* (monosyllabic word), *L* (word-initial), and *R* (word-final) respectively, but the first two characters form a known word and the last two characters do not, then the first two characters are grouped into a word and the last one is left as a monosyllabic word.

Results

The model is developed and tested using the Peking University Corpus (Yu et al. 2002). It contains all the news articles published in January, 1999 in *People's Daily*, a major newspaper in China. The corpus has 1.12 million tokens and is word-segmented. It is randomly partitioned into three parts, with 80% used for training, 10% for development, and 10% for testing. The final model is trained on the union of the training and development sets and results are reported on the test data.

As discussed earlier, two initial taggers are used in the experiment to compare the impact of the initial tagger on the TBL algorithm. In both cases, the threshold for the scoring function is set to 1, i.e., all rules that achieve two or more tagging error reductions are learned. In addition, fnTBL makes it possible to learn rules that, at the end of the training process, result in no negative application but a number of positive applications greater than a pre-determined threshold. This threshold is set to 1 as well. The results of the two initial taggers as well as the improved results achieved by the TBL algorithm are summarized in Table 2.

Tagger	Accuracy
HMM tagger	0.814
HMM + TBL	0.936
SVM tagger	0.931
SVM + TBL	0.946

Table 2: POC tagging results

The HMM tagger achieves an accuracy of 81.4%, which is improved to 93.6% by the TBL algorithm. This amounts to an absolute accuracy improvement of 12.2%, or a tagging error reduction rate of 65.6%. The SVM-based tagger achieves a better initial tagging than the HMM tagger, with an accuracy of 93.1%. The TBL algorithm achieves an absolute accuracy improvement of 1.5%, or a tagging error reduction rate of 21.7%. The better initial tagging achieved by the SVM-based tagger results in less improvement for the TBL algorithm, but better overall tagging accuracy.

The output of the merging component is evaluated with the scoring algorithm from the second SIGHAN Chinese Segmentation Bakeoff (Emerson 2005). This algorithm evaluates word segmenters in terms of recall (R), precision (P), F-score (F), recall for out-of-vocabulary words (R_{OOV}), i.e., unknown words, and recall for in-vocabulary words (R_{IV}).

The results of the model are summarized in Table 3. As the second row of the table shows, the default merger, which uses the tags assigned to the character string only, achieves an F-score of 93.5% for word segmentation with a recall rate of 73.8% for unknown word identification. Rows three through five indicate the improvement achieved by incorporating one of the three individual types of heuristics. Row three shows that the heuristics for unknown word identification (UWI) improve the recall rate for unknown words (R_{OOV}) by 2.2%; row four shows that the heuristics for long word identification (LWI) slightly improve performance on in-vocabulary words; and row five shows that the heuristics for non-word filtering (NWF) greatly improve performance on in-vocabulary words. When all the heuristics are used, the F-score for word segmentation is improved by 1.5% to 95.0%, and the recall rate for unknown word identification is improved by 1% to 74.8%.

Resources	R	P	F	R_{OOV}	R_{IV}
POC Tags	0.932	0.938	0.935	0.738	0.944
+UWI	0.933	0.939	0.936	0.760	0.944
+LWI	0.933	0.940	0.937	0.737	0.945
+NWF	0.942	0.944	0.943	0.726	0.955
+ALL	0.947	0.952	0.950	0.748	0.959

Table 3: Word segmentation results

Since the model uses no additional resources other than the training data, the results are comparable with the results of the systems that participated in the closed track of the Peking University Corpus in the second SIGHAN Chinese Segmentation Bakeoff. Table 4 summarizes the results of our model and the top three systems in the bakeoff. As these results indicate, without a complicated unknown word recognition mechanism, the final model performs at the state of the art for word segmentation along with a competitive recall rate for unknown word identification.

System	R	P	F	R_{OOV}	R_{IV}
Our model	0.947	0.952	0.950	0.748	0.959
Chen et al. (2005)	0.953	0.946	0.950	0.636	0.972
Tseng et al. (2005)	0.946	0.954	0.950	0.787	0.956
Zhang et al. (2005)	0.952	0.945	0.949	0.673	0.969

Table 4: Word segmentation results

Discussion and Conclusions

The results confirm our hypothesis that the notion of character-based tagging is useful for modeling the tendency of Chinese characters to combine with adjacent characters to form words in different contexts and has good potential for integrating unknown word identification with Chinese word segmentation. One advantage for adopting this notion is that since we have a classification problem, the model can directly benefit from improvements in classifiers. Another advantage of the two-component setup is that it allows easy integration of additional linguistic and statistical heuristics in the merging stage. The heuristics for detecting unknown words with regular internal structures prove useful for enhancing performance on unknown words. Results of the heuristics for long word recognition confirm the hypothesis that long words behave more consistently than shorter words. The heuristics for non-word filtering filter a large proportion of false unknown words detected by the tagger.

The current model does not make use of any additional resources other than the training data. Various lexical resources have been used in different word segmentation systems. For example, some of the resources used in Gao et al. (2005), which is by far the most sophisticated segmentation system, include a 98,668-entry lexicon, a morpho-lexicon that contains 59,960 morphologically derived words with information about morphological patterns and stems for each entry, as well as lists of family name characters, location names, organization names, transliterated name characters, single-character person names, and single-character location names. Such resources can be used both to improve the tagging component by enriching the feature set for tagging and to improve the heuristics in the merging component.

References

Brill, E. 1995. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics* 21(4):543–565.

Charniak, E., Hendrickson, C., Jacobson, N., and Perkowitz, M. 1993. Equations for part-of-speech tagging. In *Proceedings of AAAI 1993,* 784-789.

Chen, A., Zhou, Y., Zhang, A., and Sun, G. 2005. Unigram language model for Chinese word segmentation. In *Proceedings of the 4th SIGHAN Workshop on Chinese Language Processing*, 138-141.

Emerson, T. 2005. The Second Chinese Word Segmentation Bakeoff. In *Proceedings of the 4th SIGHAN Workshop on Chinese Language Processing*, 123-133.

Gao, J., Li, M., Wu, A., and Huang C.-N. 2005. Chinese word segmentation and named entity recognition: a pragmatic approach. *Computational Linguistics* 31(4):531–574.

Ge, X., Pratt, W., and Smyth, P. 1999. Discovering Chinese words from unsegmented text. In *Proceedings of ACM SIGIR 1999*, 271-272.

Giménez, J., and Màrquez, L. 2004. SVMTool: A general POS tagger generator based on support vector machines. In *Proceedings of LREC 2004*.

Hockenmaier, J., and Brew, C. 1998. Error-driven segmentation of Chinese. In *Proceedings of PACLIC 1998*, 218-229.

Ngai, G., and Florian, R. 2001. Transformation-based learning in the fast lane. In *Proceedings of NAACL 2001,* 40-47.

Nie, J.-Y., Hannan, M.-L., and Jin, W. 1995. Unknown word detection and segmentation of Chinese using statistical and heuristic knowledge, *Communications of COLIPS 5*(1/2), 47–57.

Rabiner, L. R. 1989. A tutorial of hidden Markov models and selected applications in speech recognition. In *Proceedings of IEEE 1989*, 257–286.

Sproat, R., Shih, C., Gale, W., and Chang, N. 1996. A stochastic finite-state word segmentation algorithm for Chinese. *Computational Linguistics 22*(3), 377–404.

Tseng, H., Chang, P., Andrew, G., Jurafsky, D., and Manning C. 2005. A conditional random fields word segmenter for SIGHAN Bakeoff 2005. In *Proceedings of the 4th SIGHAN Workshop on Chinese Language Processing*, 168-171.

Vapnik, V. N. 1995. *The Nature of Statistical Learning Theory*. Berlin: Springer-Verlag.

Xue, N. 2003. Chinese word segmentation as character tagging. *International Journal of Computational Linguistics and Chinese Language Processing* 8(1):29–48.

Yu, S., Duan, H., Zhu, X., and Sun, B. 2002. The basic processing of Contemporary Chinese Corpus at Peking University. *Journal of Chinese Information Processing* 16(5):49–64.

Zhang, H., Liu, T., Ma, J., and Liao, X. 2005. Chinese word segmentation with multiple postprocessors in HIT-IR Lab. In *Proceedings of the 4th SIGHAN Workshop on Chinese Language Processing*, 172-175.