A Trainable Rule-based Algorithm for Word Segmentation

David D. Palmer
The MITRE Corporation
202 Burlington Rd.
Bedford, MA 01730, USA
palmer@mitre.org

Abstract

This paper presents a trainable rule-based algorithm for performing word segmentation. The algorithm provides a simple, language-independent alternative to large-scale lexical-based segmenters requiring large amounts of knowledge engineering. As a stand-alone segmenter, we show our algorithm to produce high performance Chinese segmentation. In addition, we show the transformation-based algorithm to be effective in improving the output of several existing word segmentation algorithms in three different languages.

1 Introduction

This paper presents a trainable rule-based algorithm for performing word segmentation. Our algorithm is effective both as a high-accuracy stand-alone segmenter and as a postprocessor that improves the output of existing word segmentation algorithms.

In the writing systems of many languages, including Chinese, Japanese, and Thai, words are not delimited by spaces. Determining the word boundaries, thus tokenizing the text, is usually one of the first necessary processing steps, making tasks such as part-of-speech tagging and parsing possible. A variety of methods have recently been developed to perform word segmentation and the results have been published widely.[1]

A major difficulty in evaluating segmentation algorithms is that there are no widely-accepted guidelines as to what constitutes a word, and there is therefore no agreement on how to "correctly" segment a text in an unsegmented language. It is frequently mentioned in segmentation papers that native speakers of a language do not always agree about the "correct" segmentation and that the same text could be segmented into several very different (and equally correct) sets of words by different native speakers. Sproat et al.(1996) and Wu and Fung (1994) give empirical results showing that an agreement rate between native speakers as low as 75% is common. Consequently, an algorithm which scores extremely well compared to one native segmentation may score dismally compared to other, equally "correct" segmentations. We will discuss some other issues in evaluating word segmentation in Section 3.1.

One solution to the problem of multiple correct segmentations might be to establish specific guidelines for what is and is not a word in unsegmented languages. Given these guidelines, all corpora could theoretically be uniformly segmented according to the same conventions, and we could directly compare existing methods on the same corpora. While this approach has been successful in driving progress in NLP tasks such as part-of-speech tagging and parsing, there are valid arguments against adopting it for word segmentation. For example, since word segmentation is merely a preprocessing task for a wide variety of further tasks such as parsing, information extraction, and information retrieval, different segmentations can be useful or even essential for the different tasks. In this sense, word segmentation is similar to speech recognition, in which a system must be robust enough to adapt to and recognize the multiple speaker-dependent "correct" pronunciations of words. In some cases, it may also be necessary to allow multiple "correct" segmentations of the same text, depending on the requirements of further processing steps. However, many algorithms use extensive domain-specific word lists and intricate name recognition routines as well as hard-coded morphological analysis modules to produce a predetermined segmentation output. Modifying or retargeting an

[1] Most published segmentation work has been done for Chinese. For a discussion of recent Chinese segmentation work, see Sproat et al. (1996).

existing segmentation algorithm to produce a different segmentation can be difficult, especially if it is not clear what and where the systematic differences in segmentation are.

It is widely reported in word segmentation papers,[2] that the greatest barrier to accurate word segmentation is in recognizing words that are not in the lexicon of the segmenter. Such a problem is dependent both on the source of the lexicon as well as the correspondence (in vocabulary) between the text in question and the lexicon. Wu and Fung (1994) demonstrate that segmentation accuracy is significantly higher when the lexicon is constructed using the same type of corpus as the corpus on which it is tested. We argue that rather than attempting to construct a single exhaustive lexicon or even a series of domain-specific lexica, it is more practical to develop a robust trainable means of compensating for lexicon inadequacies. Furthermore, developing such an algorithm will allow us to perform segmentation in many different languages without requiring extensive morphological resources and domain-specific lexica in any single language.

For these reasons, we address the problem of word segmentation from a different direction. We introduce a rule-based algorithm which can produce an accurate segmentation of a text, given a rudimentary initial approximation to the segmentation. Recognizing the utility of multiple correct segmentations of the same text, our algorithm also allows the output of a wide variety of existing segmentation algorithms to be adapted to different segmentation schemes. In addition, our rule-based algorithm can also be used to supplement the segmentation of an existing algorithm in order to compensate for an incomplete lexicon. Our algorithm is trainable and language independent, so it can be used with any unsegmented language.

2 Transformation-based Segmentation

The key component of our trainable segmentation algorithm is Transformation-based Error-driven Learning, the corpus-based language processing method introduced by Brill (1993a). This technique provides a simple algorithm for learning a sequence of rules that can be applied to various NLP tasks. It differs from other common corpus-based methods in several ways. For one, it is weakly statistical, but not probabilistic; transformation-based approaches consequently require far less training data than most statistical approaches. It is rule-based, but relies on machine learning to acquire the rules, rather than expensive manual knowledge engineering. The rules produced can be inspected, which is useful for gaining insight into the nature of the rule sequence and for manual improvement and debugging of the sequence. The learning algorithm also considers the entire training set at all learning steps, rather than decreasing the size of the training data as learning progresses, such as is the case in decision-tree induction (Quinlan, 1986). For a thorough discussion of transformation-based learning, see Ramshaw and Marcus (1996).

Brill's work provides a proof of viability of transformation-based techniques in the form of a number of processors, including a (widely-distributed) part-of-speech tagger (Brill, 1994), a procedure for prepositional phrase attachment (Brill and Resnik, 1994), and a bracketing parser (Brill, 1993b). All of these provided performance comparable to or better than previous attempts. Transformation-based learning has also been successfully applied to text chunking (Ramshaw and Marcus, 1995), morphological disambiguation (Oflazer and Tur, 1996), and phrase parsing (Vilain and Day, 1996).

2.1 Training

Word segmentation can easily be cast as a transformation-based problem, which requires an initial model, a goal state into which we wish to transform the initial model (the "gold standard"), and a series of transformations to effect this improvement. The transformation-based algorithm involves applying and scoring all the possible rules to training data and determining which rule improves the model the most. This rule is then applied to all applicable sentences, and the process is repeated until no rule improves the score of the training data. In this manner a sequence of rules is built for iteratively improving the initial model. Evaluation of the rule sequence is carried out on a test set of data which is independent of the training data.

If we treat the output of an existing segmentation algorithm[3] as the initial state and the desired segmentation as the goal state, we can perform a series of transformations on the initial state - removing extraneous boundaries and inserting new boundaries - to obtain a more accurate approximation of the goal state. We therefore need only define an appropriate rule syntax for transforming this initial approxima-

[2] See, for example, Sproat et al. (1996).

[3] The "existing" algorithm does not need to be a large or even accurate system; the algorithm can be arbitrarily simple as long as it assigns some form of initial segmentation.

tion and prepare appropriate training data.

For our experiments, we obtained corpora which had been manually segmented by native or near-native speakers of Chinese and Thai. We divided the hand-segmented data randomly into training and test sets. Roughly 80% of the data was used to train the segmentation algorithm, and 20% was used as a blind test set to score the rules learned from the training data. In addition to Chinese and Thai, we also performed segmentation experiments using a large corpus of English in which all the spaces had been removed from the texts. Most of our English experiments were performed using training and test sets with roughly the same 80-20 ratio, but in Section 3.4.3 we discuss results of English experiments with different amounts of training data. Unfortunately, we could not repeat these experiments with Chinese and Thai due to the small amount of hand-segmented data available.

2.2 Rule syntax

There are three main types of transformations which can act on the current state of an imperfect segmentation:

- Insert - place a new boundary between two characters
- Delete - remove an existing boundary between two characters
- Slide - move an existing boundary from its current location between two characters to a location 1, 2, or 3 characters to the left or right[4]

In our syntax, Insert and Delete transformations can be triggered by any two adjacent characters (a bigram) and one character to the left or right of the bigram. Slide transformations can be triggered by a sequence of one, two, or three characters over which the boundary is to be moved. Figure 1 enumerates the 22 segmentation transformations we define.

3 Results

With the above algorithm in place, we can use the training data to produce a rule sequence to augment an initial segmentation approximation in order to obtain a better approximation of the desired segmentation. Furthermore, since all the rules are purely character-based, a sequence can be learned for any character set and thus any language. We used our rule-based algorithm to improve the word segmentation rate for several segmentation algorithms in three languages.

[4]Note that a Slide transformation is equivalent to a Delete plus an Insert.

3.1 Evaluation of segmentation

Despite the number of papers on the topic, the evaluation and comparison of existing segmentation algorithms is virtually impossible. In addition to the problem of multiple correct segmentations of the same texts, the comparison of algorithms is difficult because of the lack of a single metric for reporting scores. Two common measures of performance are *recall* and *precision*, where recall is defined as the percent of words in the hand-segmented text identified by the segmentation algorithm, and precision is defined as the percentage of words returned by the algorithm that also occurred in the hand-segmented text in the same position. The component recall and precision scores are then used to calculate an *F-measure* (Rijsbergen, 1979), where $F = (1 + \beta)PR/(\beta P + R)$. In this paper we will report all scores as a balanced F-measure (precision and recall weighted equally) with $\beta = 1$, such that $F = 2PR/(P + R)$

3.2 Chinese

For our Chinese experiments, the training set consisted of 2000 sentences (60187 words) from a Xinhua news agency corpus; the test set was a separate set of 560 sentences (18783 words) from the same corpus.[5] We ran four experiments using this corpus, with four different algorithms providing the starting point for the learning of the segmentation transformations. In each case, the rule sequence learned from the training set resulted in a significant improvement in the segmentation of the test set.

3.2.1 Character-as-word (CAW)

A very simple initial segmentation for Chinese is to consider each character a distinct word. Since the average word length is quite short in Chinese, with most words containing only 1 or 2 characters,[6] this character-as-word segmentation correctly identified many one-character words and produced an initial segmentation score of F=40.3. While this is a low segmentation score, this segmentation algorithm identifies enough words to provide a reasonable initial segmentation approximation. In fact, the CAW algorithm alone has been shown (Buckley et al., 1996; Broglio et al., 1996) to be adequate to be used successfully in Chinese information retrieval.

Our algorithm learned 5903 transformations from the 2000 sentence training set. The 5903 transformations applied to the test set improved the score from F=40.3 to 78.1, a 63.3% reduction in the error

[5]The Chinese texts were prepared by Tom Keenan.

[6]The average length of a word in our Chinese data was 1.60 characters.

Rule	Boundary Action	Triggering Context
AB ⟺ A B	Insert (delete) between A and B	any
xB ⟺ x B	Insert (delete) before any B	any
Ay ⟺ A y	Insert (delete) after any A	any
ABC ⟺ A B C	Insert (delete) between A and B AND Insert (delete) between B and C	any
JAB ⟺ JA B	Insert (delete) between A and B	J to left of A
¬JAB ⟺ ¬JA B	Insert (delete) between A and B	no J to left of A
ABK ⟺ A BK	Insert (delete) between A and B	K to right of B
AB¬K ⟺ A B¬K	Insert (delete) between A and B	no K to right of B
xA y ⟺ x Ay	Move from after A to before A	any
xAB y ⟺ x ABy	Move from after bigram AB to before AB	any
xABC y ⟺ x ABCy	Move from after trigram ABC to before ABC	any

Figure 1: Possible transformations. A, B, C, J, and K are specific characters; x and y can be any character. ¬J and ¬K can be any character except J and K, respectively.

rate. This is a very surprising and encouraging result, in that, from a very naive initial approximation using no lexicon except that implicit from the training data, our rule-based algorithm is able to produce a series of transformations with a high segmentation accuracy.

3.2.2 Maximum matching (greedy) algorithm

A common approach to word segmentation is to use a variation of the *maximum matching* algorithm, frequently referred to as the "greedy algorithm." The greedy algorithm starts at the first character in a text and, using a word list for the language being segmented, attempts to find the longest word in the list starting with that character. If a word is found, the maximum-matching algorithm marks a boundary at the end of the longest word, then begins the same longest match search starting at the character following the match. If no match is found in the word list, the greedy algorithm simply skips that character and begins the search starting at the next character. In this manner, an initial segmentation can be obtained that is more informed than a simple character-as-word approach. We applied the maximum matching algorithm to the test set using a list of 57472 Chinese words from the NMSU CHSEG segmenter (described in the next section). This greedy algorithm produced an initial score of F=64.4.

A sequence of 2897 transformations was learned from the training set; applied to the test set, they improved the score from F=64.4 to 84.9, a 57.8% error reduction. From a simple Chinese word list, the rule-based algorithm was thus able to produce a segmentation score comparable to segmentation algorithms developed with a large amount of domain knowledge (as we will see in the next section).

This score was improved further when combining the character-as-word (CAW) and the maximum matching algorithms. In the maximum matching algorithm described above, when a sequence of characters occurred in the text, and no subset of the sequence was present in the word list, the entire sequence was treated as a single word. This often resulted in words containing 10 or more characters, which is very unlikely in Chinese. In this experiment, when such a sequence of characters was encountered, each of the characters was treated as a separate word, as in the CAW algorithm above. This variation of the greedy algorithm, using the same list of 57472 words, produced an initial score of F=82.9. A sequence of 2450 transformations was learned from the training set; applied to the test set, they improved the score from F=82.9 to 87.7, a 28.1% error reduction. The score produced using this variation of the maximum matching algorithm combined with a rule sequence (87.7) is nearly equal to the score produced by the NMSU segmenter segmenter (87.9) discussed in the next section.

3.2.3 NMSU segmenter

The previous three experiments showed that our rule sequence algorithm can produce excellent segmentation results given very simple initial segmentation algorithms. However, assisting in the adaptation of an existing algorithm to different segmentation schemes, as discussed in Section 1, would most likely be performed with an already accurate, fully-developed algorithm. In this experiment we demon-

strate that our algorithm can also improve the output of such a system.

The Chinese segmenter CHSEG developed at the Computing Research Laboratory at New Mexico State University is a complete system for high-accuracy Chinese segmentation (Jin, 1994). In addition to an initial segmentation module that finds words in a text based on a list of Chinese words, CHSEG additionally contains specific modules for recognizing idiomatic expressions, derived words, Chinese person names, and foreign proper names. The accuracy of CHSEG on an 8.6MB corpus has been independently reported as F=84.0 (Ponte and Croft, 1996). (For reference, Ponte and Croft report scores of F=86.1 and 83.6 for their probabilistic Chinese segmentation algorithms trained on over 100MB of data.)

On our test set, CHSEG produced a segmentation score of F=87.9. Our rule-based algorithm learned a sequence of 1755 transformations from the training set; applied to the test set, they improved the score from 87.9 to 89.6, a 14.0% reduction in the error rate. Our rule-based algorithm is thus able to produce an improvement to an existing high-performance system.

Table 1 shows a summary of the four Chinese experiments.

3.3 Thai

While Thai is also an unsegmented language, the Thai writing system is alphabetic and the average word length is greater than Chinese.[7] We would therefore expect that our character-based transformations would not work as well with Thai, since a context of more than one character is necessary in many cases to make many segmentation decisions in alphabetic languages.

The Thai corpus consisted of texts[8] from the Thai News Agency via NECTEC in Thailand. For our experiment, the training set consisted of 3367 sentences (40937 words); the test set was a separate set of 1245 sentences (13724 words) from the same corpus.

The initial segmentation was performed using the maximum matching algorithm, with a lexicon of 9933 Thai words from the word separation filter in *cttex*, a Thai language Latex package. This greedy algorithm gave an initial segmentation score of F=48.2 on the test set.

[7]The average length of a word in our Thai data was 5.01 characters.

[8]The Thai texts were manually segmented by Jo Tyler.

Our rule-based algorithm learned a sequence of 731 transformations which improved the score from 48.2 to 63.6, a 29.7% error reduction. While the alphabetic system is obviously harder to segment, we still see a significant reduction in the segmenter error rate using the transformation-based algorithm. Nevertheless, it is doubtful that a segmentation with a score of 63.6 would be useful in too many applications, and this result will need to be significantly improved.

3.4 De-segmented English

Although English is not an unsegmented language, the writing system is alphabetic like Thai and the average word length is similar.[9] Since English language resources (e.g. word lists and morphological analyzers) are more readily available, it is instructive to experiment with a de-segmented English corpus, that is, English texts in which the spaces have been removed and word boundaries are not explicitly indicated. The following shows an example of an English sentence and its de-segmented version:

> About 20,000 years ago the last ice age ended.
>
> About20,000yearsagothelasticeageended.

The results of such experiments can help us determine which resources need to be compiled in order to develop a high-accuracy segmentation algorithm in unsegmented alphabetic languages such as Thai. In addition, we are also able to provide a more detailed error analysis of the English segmentation (since the author can read English but not Thai).

Our English experiments were performed using a corpus of texts from the Wall Street Journal (WSJ). The training set consisted of 2675 sentences (64632 words) in which all the spaces had been removed; the test set was a separate set of 700 sentences (16318 words) from the same corpus (also with all spaces removed).

3.4.1 Maximum matching experiment

For an initial experiment, segmentation was performed using the maximum matching algorithm, with a large lexicon of 34272 English words compiled from the WSJ.[10] In contrast to the low initial Thai score, the greedy algorithm gave an initial English segmentation score of F=73.2. Our rule-based algorithm learned a sequence of 800 transformations,

[9]The average length of a word in our English data was 4.46 characters, compared to 5.01 for Thai and 1.60 for Chinese.

[10]Note that the portion of the WSJ corpus used to compile the word list was independent of both the training and test sets used in the segmentation experiments.

Initial algorithm	Initial score	Rules learned	Improved score	Error reduction
Character-as-word	40.3	5903	78.1	63.3%
Maximum matching	64.4	2897	84.9	57.8%
Maximum matching + CAW	82.9	2450	87.7	28.1%
NMSU segmenter	87.9	1755	89.6	14.0%

Table 1: Chinese results.

which improved the score from 73.2 to 79.0, a 21.6% error reduction.

The difference in the greedy scores for English and Thai demonstrates the dependence on the word list in the greedy algorithm. For example, an experiment in which we randomly removed half of the words from the English list reduced the performance of the greedy algorithm from 73.2 to 32.3; although this reduced English word list was nearly twice the size of the Thai word list (17136 vs. 9939), the longest match segmentation utilizing the list was much lower (32.3 vs. 48.2). Successive experiments in which we removed different random sets of half the words from the original list resulted in greedy algorithm performance of 39.2, 35.1, and 35.5. Yet, despite the disparity in initial segmentation scores, the transformation sequences effect a significant error reduction in all cases, which indicates that the transformation sequences are effectively able to compensate (to some extent) for weaknesses in the lexicon. Table 2 provides a summary of the results using the greedy algorithm for each of the three languages.

3.4.2 Basic morphological segmentation experiment

As mentioned above, lexical resources are more readily available for English than for Thai. We can use these resources to provide an informed initial segmentation approximation separate from the greedy algorithm. Using our native knowledge of English as well as a short list of common English prefixes and suffixes, we developed a simple algorithm for initial segmentation of English which placed boundaries after any of the suffixes and before any of the prefixes, as well as segmenting punctuation characters. In most cases, this simple approach was able to locate only one of the two necessary boundaries for recognizing full words, and the initial score was understandably low, F=29.8. Nevertheless, even from this flawed initial approximation, our rule-based algorithm learned a sequence of 632 transformations which nearly doubled the word recall, improving the score from 29.8 to 53.3, a 33.5% error reduction.

3.4.3 Amount of training data

Since we had a large amount of English data, we also performed a classic experiment to determine the effect the amount of training data had on the ability of the rule sequences to improve segmentation. We started with a training set only slightly larger than the test set, 872 sentences, and repeated the maximum matching experiment described in Section 3.4.1. We then incrementally increased the amount of training data and repeated the experiment. The results, summarized in Table 3, clearly indicate (not surprisingly) that more training sentences produce both a longer rule sequence and a larger error reduction in the test data.

Training sentences	Rules learned	Improved score	Error reduction
872	436	78.2	18.9%
1731	653	78.9	21.3%
2675	800	79.0	21.6%
3572	902	79.4	23.1%
4522	1015	80.3	26.5%

Table 3: English training set sizes. Initial score of test data (700 sentences) was 73.2.

3.4.4 Error analysis

Upon inspection of the English segmentation errors produced by both the maximum matching algorithm and the learned transformation sequences, one major category of errors became clear. Most apparent was the fact that the limited context transformations were unable to recover from many errors introduced by the naive maximum matching algorithm. For example, because the greedy algorithm always looks for the longest string of characters which can be a word, given the character sequence "economicsituation", the greedy algorithm first recognized "economics" and several shorter words, segmenting the sequence as "economics it u at io n". Since our transformations consider only a single character of context, the learning algorithm was unable to patch the smaller segments back together to produce the desired output "economic situation". In some cases,

Language	Lexicon size	Initial score	Rules learned	Improved score	Error reduction
Chinese	57472	64.4	2897	84.9	57.8%
Chinese (with CAW)	57472	82.9	2450	87.7	28.1%
Thai	9939	48.2	731	63.6	29.7%
English	34272	73.2	800	79.0	21.6%

Table 2: Summary of maximum matching results.

the transformations were able to recover some of the word, but were rarely able to produce the full desired output. For example, in one case the greedy algorithm segmented "humanactivity" as "humana c ti vi ty". The rule sequence was able to transform this into "humana ctivity", but was not able to produce the desired "human activity". This suggests that both the greedy algorithm and the transformation learning algorithm need to have a more global word model, with the ability to recognize the impact of placing a boundary on the longer sequences of characters surrounding that point.

4 Discussion

The results of these experiments demonstrate that a transformation-based rule sequence, supplementing a rudimentary initial approximation, can produce accurate segmentation. In addition, they are able to improve the performance of a wide range of segmentation algorithms, without requiring expensive knowledge engineering. Learning the rule sequences can be achieved in a few hours and requires no language-specific knowledge. As discussed in Section 1, this simple algorithm could be used to adapt the output of an existing segmentation algorithm to different segmentation schemes as well as compensating for incomplete segmenter lexica, without requiring modifications to segmenters themselves.

The rule-based algorithm we developed to improve word segmentation is very effective for segmenting Chinese; in fact, the rule sequences combined with a very simple initial segmentation, such as that from a maximum matching algorithm, produce performance comparable to manually-developed segmenters. As demonstrated by the experiment with the NMSU segmenter, the rule sequence algorithm can also be used to improve the output of an already highly-accurate segmenter, thus producing one of the best segmentation results reported in the literature.

In addition to the excellent overall results in Chinese segmentation, we also showed the rule sequence algorithm to be very effective in improving segmentation in Thai, an alphabetic language. While the scores themselves were not as high as the Chinese performance, the error reduction was nevertheless very high, which is encouraging considering the simple rule syntax used. The current state of our algorithm, in which only three characters are considered at a time, will understandably perform better with a language like Chinese than with an alphabetic language like Thai, where average word length is much greater. The simple syntax described in Section 2.2 can, however, be easily extended to consider larger contexts to the left and the right of boundaries; this extension would necessarily come at a corresponding cost in learning speed since the size of the rule space searched during training would grow accordingly. In the future, we plan to further investigate the application of our rule-based algorithm to alphabetic languages.

Acknowledgements This work would not have been possible without the assistance and encouragement of all the members of the MITRE Natural Language Group. This paper benefited greatly from discussions with and comments from Marc Vilain, Lynette Hirschman, Sam Bayer, and the anonymous reviewers.

References

Eric Brill and Philip Resnik. 1994. A rule-based approach to prepositional phrase attachment disambiguation. In *Proceedings of the Fifteenth International Conference on Computational Linguistics (COLING-1994).*

Eric Brill. 1993a. A corpus-based approach to language learning. Ph.D. Dissertation, University of Pennsylvania, Department of Computer and Information Science.

Eric Brill. 1993b. Transformation-based error-driven parsing. In *Proceedings of the Third International Workshop on Parsing Technologies.*

Eric Brill. 1994. Some advances in transformation-based part of speech tagging. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 722–727.

John Broglio, Jamie Callan, and W. Bruce Croft. 1996. Technical issues in building an information retrieval system for chinese. CIIR Technical Report IR-86, University of Massachusetts, Amherst.

Chris Buckley, Amit Singhal, and Mandar Mitra. 1996. Using query zoning and correlation within smart: Trec 5. In *Proceedings of the Fifth Text Retrieval Conference (TREC-5).*

Wanying Jin. 1994. Chinese segmentation disambiguation. In *Proceedings of the Fifteenth International Conference on Computational Linguistics (COLING-94)*, Japan.

Judith L. Klavans and Philip Resnik. 1996. *The Balancing Act: Combining Symbolic and Statistical Approaches to Language.* MIT Press, Cambridge, MA.

Kemal Oflazer and Gokhan Tur. 1996. Combining hand-crafted rules and unsupervised learning in constraint-based morphological disambiguation. In *Proceedings of the Conference on Empirical Methods in Language Processing (EMNLP).*

Jay M. Ponte and W. Bruce Croft. 1996. Useg: A retargetable word segmentation procedure for information retrieval. In *Proceedings of SDAIR96*, Las Vegas, Nevada.

J.R. Quinlan. 1986. Induction of decision trees. *Machine Learning*, 1(1):81–106.

Lance Ramshaw and Mitchell Marcus. 1995. Text chunking using transformation-based learning. In *Proceedings of the Third Workshop on Very Large Corpora (WVLC-3)*, pages 82–94.

Lance A. Ramshaw and Mitchell P. Marcus. 1996. Exploring the nature of transformation-based learning. In *Klavans and Resnik (1996).*

C. J. Van Rijsbergen. 1979. *Information Retrieval.* Butterworths, London.

Giorgio Satta and Eric Brill. 1996. Efficient transformation-based parsing. In *Proceedings of the Thirty-fourth Annual Meeting of the Association for Computational Linguistics (ACL-96).*

Richard W. Sproat, Chilin Shih, William Gale, and Nancy Chang. 1996. A stochastic finite-state word-segmentation algorithm for chinese. *Computational Linguistics*, 22(3):377–404.

Marc Vilain and David Day. 1996. Finite-state phrase parsing by rule sequences. In *Proceedings of the Sixteenth International Conference on Computational Linguistics (COLING-96).*

Marc Vilain and David Palmer. 1996. Transformation-based bracketing: Fast algorithms and experimental results. In *Proceedings of the Workshop on Robust Parsing, held at ESSLLI 1996.*

Dekai Wu and Pascale Fung. 1994. Improving chinese tokenization with linguistic filters on statistical lexical acquisition. In *Proceedings of the Fourth ACL Conference on Applied Natural Language Processing (ANLP94)*, Stuttgart, Germany.

Zimin Wu and Gwyneth Tseng. 1993. Chinese text segmentation for text retrieval: Achievements and problems. *Journal of the American Society for Information Science*, 44(9):532–542.