Chinese Word Segmentation for Terrorism-Related Contents

Daniel Zeng[1,2], Donghua Wei[1], Michael Chau[3], and Feiyue Wang[1,2]

[1] Institute of Automation, Chinese Academy of Sciences, China
[2] The University of Arizona, Tucson, Arizona, USA
[3] The University of Hong Kong, Hong Kong
zeng@email.arizona.edu, donghuawei@gmail.com,
mchau@business.hku.hk, feiyue.wang@ia.ac.cn

Abstract. In order to analyze security and terrorism related content in Chinese, it is important to perform word segmentation on Chinese documents. There are many previous studies on Chinese word segmentation. The two major approaches are statistic-based and dictionary-based approaches. The pure statistic methods have lower precision, while the pure dictionary-based method cannot deal with new words and are restricted to the coverage of the dictionary. In this paper, we propose a hybrid method that avoids the limitations of both approaches. Through the use of suffix tree and mutual information (MI) with the dictionary, our segmenter, called IASeg, achieves a high accuracy in word segmentation when domain training is available. It can identify new words through MI-based token merging and dictionary update. In addition, with the Improved Bigram method it can also process N-grams. To evaluate the performance of our segmenter, we compare it with the Hylanda segmenter and the ICTCLAS segmenter using a terrorism-related corpus. The experiment results show that IASeg performs better than the two benchmarks in both precision and recall.

Keywords: Mutual information, N-gram, suffix tree, Ukkonen algorithm, Heuristic rules, Lidstone flatness.

1 Introduction

Extremists and terrorists have been using the Internet to spread their ideology and recruit new members. It is important for government and anti-terrorist organizations to analyze such online information in order to enhance national and international security. Previous research has reported on how to collect and analyze relevant documents from the Internet (e.g., Web pages, blogs, newsgroup postings) in English and it has been shown possible to extract extremist or terrorist information and their relationships from these documents (Chen & Xu, 2006; Chau & Xu, 2007). However, little work has been done in the analysis of extremist or terrorist related Web documents in Chinese. In this paper we propose a method that combines mutual information and suffix tree to address the word segmentation problem in Chinese document analysis. We apply the proposed method to perform word segmentation on a terrorism-related corpus. The rest of the paper is structured as follows. Section 2 reviews related work in Chinese word segmentation. In Section 3 we describe our proposed algorithm based on mutual information and suffix tree. Section 4 reports the results of our evaluation study, in which we tested our proposed algorithm using a terrorism-related data set. We discuss the findings in Section 5 and conclude our study in Section 6.

2 Related Work

Chinese word segmentation has been studied for many years, but two problems in word segmentation, namely unknown word identification and ambiguity parsing, are still not completely solved. Studies on Chinese word segmentation can be roughly divided into two categories: heuristic dictionary-based methods, and statistical machine learning methods. Readers are referred to (Wu et al., 1993) for a more detail survey. In the following, we review previous research in each category.

2.1 Dictionary-Based Methods

Dictionary-based methods mainly employ a predefined dictionary and some hand-generated rules for segmenting input sequence. These rules can be generally classified based on the scanning direction and the prior matching length. The Forward Matching Method (FMM), the input string will be scanned from the beginning to the end and matched against dictionary entries. In the Reverse Matching Method (RMM), the input string will be scanned from the end to the beginning. Bidirectional Matching Methods scan the input string from both directions. The matching length can be based on maximum matching or minimum matching. Most popular dictionary-based segmenters use a hybrid matching method. The main disadvantage of dictionary-based methods is that their performance depends on the coverage of the lexicon, which unfortunately may never be complete because new words appear constantly. Consequently, these methods cannot deal with the unknown words (sometimes called Out-Of-Vocabulary or OOV) identification and may result in wrong segmentation.

2.2 Statistical and Machine Learning Methods

Statistical methods rely on different measure to decide on the segmentation boundary in Chinese. Sun et al.(2004) use a liner function of mutual information (MI) and difference of t-test to perform text segmentation. Many researchers also concentrate on two-character words (bigrams), because two is the most common length in Chinese words. Dai et al. (1999) use contextual and positional information, and found that contextual information is the most important factor for bigram extraction. They found that positional frequency is not helpful in determining words. Yu et al. (2006) proposed a cascaded Hidden Markov Model (HMM) for location and organization identification. Other researchers, such as Jia et al. (2007), Xue et al. (2003), and Low et al. (2005) focus on the Maximum Entropy (ME) models. Li et al. (2002) use Expectation Maximization and Maximum Likelihood Prediction to deal with Chinese word segmentation. Zhou & Liu (2002) construct state chart using the information of whether several characters can compose one word and propose an algorithm to generate candidate words. Sproat et al. (1996) propose a Stochastic Finite-State Word-Segmentation method by combining character states with dictionary-based heuristic rules. There are several others: Hockenmaier and Brew (1998) present an algorithm, based on Palmer's (1997) experiments, that applies a symbolic machine learning technique to the problem of Chinese word segmentation. Many other statistic-based machine learning methods have been used in Chinese word segmentation, such as SVM-based segmentation (Li et al., 2001), the CRF method segmentation (Peng, et al., 2004), unsupervised models (Creutz et al, 2007).

Ponte and Croft (1996) introduce two models for word segmentation: word-based and bigram-based models. Both utilize probabilistic automata. In the word-based method, a suffix tree of words in the lexicon is used to initialize the model. Each node is associated with a probability, which is estimated by segmenting training text using the longest match strategy. This makes it easy to apply the segmenter to new languages. The bigram model uses the lexicon to initialize probability estimates for each bigram, and the probability with which each bigram occurs, and uses the Baum-Welch algorithm (Rabiner 1989) to update the probabilities as the training text is processed.

Some researchers concentrate on the named entity identification, such as Sproat et al. (1996), who developed special-purpose recognizers for Chinese names (and translated foreign names). They implemented special recognizers not only for Chinese names and transliterated foreign names, but also for components of morphologically obtained words.

As we know, pure dictionary-based methods rely on the coverage of the dictionaries, and general statistical methods require segmented training corpus. Teahan et al. (2000) proposed using text compression model to do word segmentation. This method uses neither manual dictionary nor training corpus, but just uses finite-context models of characters to predict the upcoming one. Each prediction takes the form of a probability distribution that is provided to an encoder. The conditional probability distribution of characters, conditioned on the preceding few characters, is maintained and updated as each character of input is processed.

In sum, all methods have to rely on either character-level information indicated by the co-occurrence probability, conditional probability, position status, or word-level information provided by the dictionary or the language knowledge, such as the part-of-speech, morphological, syntactic and semantic knowledge (Cui et al., 2006). Many researchers combine the available information and achieved better performance in both unsupervised learning (Peng and Schuurmans, 2001) and supervised learning (Teahan et al., 2000).

3 Proposed Algorithm

In this paper, Mutual Information (MI) and Suffix Tree are combined to perform Chinese word segmentation. While Ponte and Croft (1996) just deal with bigrams, we focus more on segmentation of trigrams and longer words. We first use a training corpus to train the bigram model and use a lexicon to establish the improved bigram model. We then use MI and the improved bigram model combining with the Suffix Tree to parse the given text.

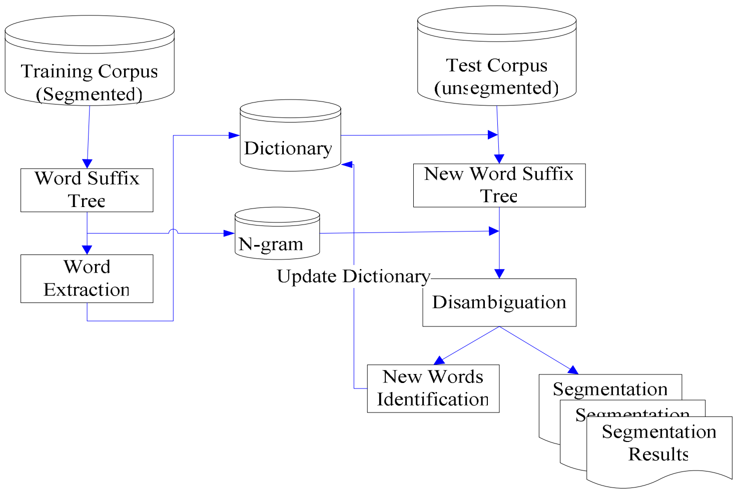


Fig. 1. Overall architecture of our system

In this Section, we describe our proposed algorithm, called the IASeg. We separate the segmentation process into two phases - the training phase and the test phase. The overall algorithm is shown in Figure 1. In the training phase, we construct a dictionary and the N-grams, which include the Unigram, Bigram and the Improved Bigram. In the test phase, we first split the input string into tokens using dictionary-based FMM heuristic. Then we calculate the strings' Mutual Information to predict unknown words and decide whether we should merge the two adjacent tokens as one new word. If the formation of the new word is supported, we can update the dictionary dynamically. Finally, we output the segmentation results.

3.1 N-Gram Construction

The n-gram word model is based on the Markovian assumption. If the *n*-th character is related only with its preceding (*N*-1) characters of context, and without any other correlations, we call it the N-gram word model or (*N*-1)-order Markov model.

Given a string $w_1w_2\ldots w_n$, with its length being n, we have the following equations.

In unigram, we have.:

$$p(w_1w_2\ldots w_n)= p(w_1)\, p(w_2)\ldots\, p(w_n)$$

Using bigram (1-order Markov model), we have:

$$p(w_1w_2\ldots w_n)= p(w_1)\, p(w_2\,|\,w_1)\, p(w_3\,|\,w_2)\ldots p(w_n\,|\,w_{n-1})$$

And using trigram (2-order Markov model), we have:

$$p(w_1w_2\ldots w_n)= p(w_1)\, p(w_2\,|\,w_1)\, p(w_3\,|\,w_1w_2)\, p(w_4\,|\,w_2w_3)\ldots p(w_n\,|\,w_{n-2}\, w_{n-1})$$

For an N-gram model, we have:

$$p(w_1w_2\ldots w_n)= p(w_1)\, p(w_2\,|\,w_1)\, p(w_3\,|\,w_1w_2)\, p(w_4\,|\,w_1w_2w_3)\ldots p(w_n\,|\,w_1\ldots w_{n-1})$$

The above equations can be expressed by one equation:

$$p(w_1w_2\ldots w_n)= \prod_{i=1}^{n} p(w_i\,|\,w_{i-(N-1)}\, w_{i-(N-1)+1}\, w_{i-(N-1)+2}\ldots w_{i-1})$$

Table 1. Expressions in equations

Expression	Meaning
w_i	a character
$length(str)$	Number of characters in *str*
w_i^j	simple expression of string of $w_iw_{i+1}...w_j$, $i < j$
$p\ (w)$	probability of string w in a given corpus
$count(w_1w_2...w_n)$	frequency of n-gram $w_1w_2...w_n$ in a given corpus
$p(x, y)$	probability of co-occurrence of x, y
$f(x)$	frequency estimate of x
N	number of training instances

where $n= length(w_1w_2...w_n)$, and N is the length of N-grams to be considered. Table 1 gives a summary of the expressions used.

To make calculation simpler we often use the following parameter evaluation equations, based on their relative frequency, using a Maximum Likelihood Estimation (MLE) method.

$$p_{MLE}\,(w_n/w_{n-1}) = \frac{count\ (w_{n-1}w_n)}{count\ (w_{n-1})}$$

$$p_{MLE}\,(w_n/w_1^{n-1}) = \frac{count(w_1w_2w_3...w_n)}{count(w_1w_2w_3...w_{n-1})}$$

$$p_{MLE}(w_n/w_{n-N+1}^{n-1}) = \frac{count(w_{n-N+1}^{n-1}w_n)}{count(w_{n-N+1}^{n-1})}$$

For an N-gram model a large number of parameters need to be estimated. Many previous studies have focused on bigrams only because of computational efficiency considerations. In this study, we use an Improved Bigram to deal with longer grams. More details will be discussed in the following subsection.

3.2 MI Measure

3.2.1 Basic Concepts and MI Calculation Equation

The concept of Mutual Information comes from <Information Theory>, which indicates two events' dependence (compactness) using:

$$MI(x, y) = log_2 \frac{p(x, y)}{p(x)p(y)}$$

In Natural Language Processing (NLP), $MI(x,y)$ is used to estimate the *compactness* of two characters: x, y. If $MI(x,y)$ is higher than a given threshold value (usually estimated through experiments, denoted μ), we can regard them as one word. To simplify calculation, we define $p_{\mathrm{MLE}}(x) = f(x) = count(x)/N$, so we can rewrite the MI equation as follows:

$$MI(x, y) = log_2 \frac{count(x, y) \cdot N}{count(x)count(y)}$$

Its full meanings are as followings: (On condition that the characters *a*, *b* both have the normal distribution in text, their dependence equals to their correlation.)

1. If $MI(a, b) >> 0$, i.e. $count(a, b) \cdot N >> count(a) \cdot count(b)$, the characters have positive correlation. And if $MI(a, b)>\mu$, we can regard the string '*ab*' as a word.

2. If $MI(a, b) << 0$, i.e. $count(a, b) \cdot N << count(a) \cdot count(b)$, the characters have negative correlation, and we do not regard '*ab*' as a word.

3. If $MI(a, b) \approx 0$, i.e. $count(a, b) \cdot N \approx count(a) \cdot count(b)$, then we say the characters have no correlation and they can't be viewed as a word either.

Researchers(Fang et al., 2005) also have used the following equation to calculate the *MI* of two bigrams *c* and *d*, each of them being a bigram string, and achieve better performance in such cases where a 4-character word is composed of 2 bigrams, e.g., "高音喇叭", in which both "高音" and "喇叭" are words.

$$MI(c,d) = log_2 \frac{N_c^2 f(c,d)}{N_w \times f(c) \times f(d)}$$

where N_c is the total characters in the corpus, N_w is the total number of the tokens in the corpus.

In our approach, however, we do not adopt this method because we have different classes of n-grams but just use one threshold. In order to come up with one measure standard, we using MLE calculation equations together with the *Lidstone* flatness algorithm to avoid the sparseness of the co-occurrences.

3.2.2 Flatness Algorithm

Most of the probabilities involved in the MI calculation are very small and can result in *zero* probability. To avoid numerical problems associated with these zero probabilities, we use the Lidstone flatness function:

$$P_{Lid}(w_1 ... w_n) = \frac{count(w_1 ... w_n) + \lambda)}{N + \lambda * B}$$

where $\lambda = 0.5$, *B* is the number of bins that the training instances are divided into (usually based on the number of dictionary items), and *N* is the corpus size (number of tokens).

For the forecast, we use

$$P(b/a) = \frac{count(a,b) + \lambda}{(count(a) + \lambda B)}, 0 < \lambda < 1$$

Especially, to deal with the probability of single word *w*, using following:

$$P(w) = \frac{count(w) + \lambda}{N + \lambda B}, 0 < \lambda < 1$$

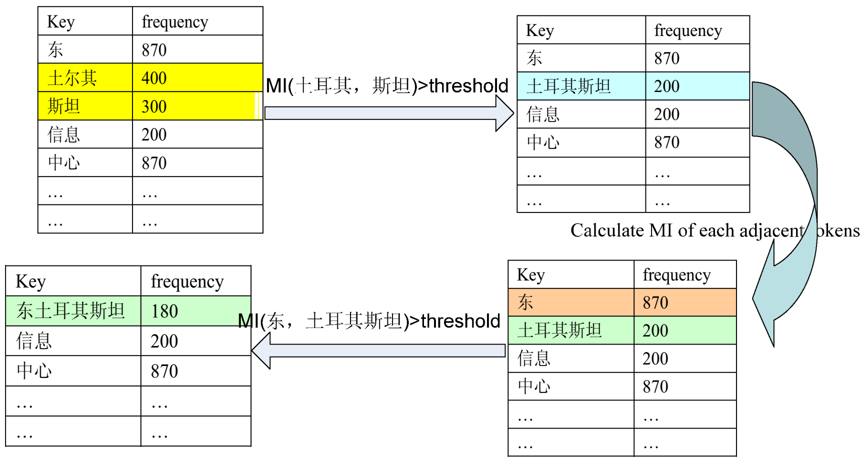
on condition that *a*, *b*, *w* are non-empty strings.

In our research, we store the tokens and their frequencies. If we need to calculate their MI, we first retrieve the tokens and their frequencies, then calculate the MI using the equations described earlier.

3.2.3 Improved Bigram

In this subsection we describe our improved bigram model which is used to deal with n-gram words with $n \geq 3$. We store patterns using a hash table for MI computation with the unigram and simple bigram. This approach allows us to process multi-gram words, such as 4-gram words and 5-gram words, and even more parameters prediction.

We show an example of our improved Bigram in Figure 2. Consider the words: "东/土耳其/斯坦/信息/中心 (East Turkistan Information Center)". The MI of "土耳其" and "斯坦" will be calculated. As this is greater than the threshold, the frequency of the term "土耳其斯坦" will be stored and the MI of "东" and "土耳其斯坦" will be further calculated to obtain the correct term "东土耳其斯坦(East Turkistan) ".


MI(土耳其，斯坦)>threshold
Calculate MI of each adjacent tokens
MI(东，土耳其斯坦)>threshold

Key	frequency
东	870
土尔其	400
斯坦	300
信息	200
中心	870
…	…
…	…

Key	frequency
东	870
土耳其斯坦	200
信息	200
中心	870
…	…
…	…

Key	frequency
东	870
土耳其斯坦	200
信息	200
中心	870
…	…
…	…

Key	frequency
东土耳其斯坦	180
信息	200
中心	870
…	…
…	…

Fig. 2. Example of Improved Bigrams

This method is useful for segmenting terms that are named entities, such as combining "邓/小平" to "邓小平(Deng Xiaopeng)", and "中国/人民/银行" to "中国人民银行 (The People's Bank of China) ", etc. It can also deal with some ambiguous pairs, such as "新西兰花". For instance, if the training corpus are mostly about vegetables (i.e., having a lot of individual occurrences of "西兰花"), then it would split into "新/西兰花 (new broccoli)"; but if the corpus are about the country (i.e., having a lot of individual occurrences of "新西兰"), then it would be segmented as "新西兰/花 (New Zealand flowers)".

3.3 Suffix Tree

Given a string $S \in \Sigma^n$, the suffix tree T_S of S is the compacted Trie of all the suffixes of $S\$$, $\$ \notin \Sigma$. The suffix tree is the basic data structure in combinatorial pattern matching because of its many elegant uses. Furthermore, it has a compact $O(n)$ space representation that can be constructed in $O(n)$ optimal time for a constant-size alphabet (Weiner 1973). The original construction and its analysis are nontrivial.

Some efforts have been spent on producing simplified linear time algorithms (Chen and Seiferas 1985; McCreight 1976), though all such efforts have been variants of the original approach of Weiner. Due to limited space, readers are referred to other papers for the details of the details of the model and implementation of suffix tree (Chen and Seiferas 1985; Giegerich, et al., 1997; McCreight 1976; Weiner 1973; Zhang et al, 2004).

Suffix Tree has a low complexity (Chan et al., 2005). Let $C=\{T_1, T_2, ..., T_k\}$ be a collection of texts with total length n. We can maintain a compressed suffix tree for C, which uses $O(n)$ space and supports the following queries about the suffix tree for C: finding the root in $O(1)$ time and finding the parent, left child, left sibling, right sibling, and suffix link of a node in $O(\log n)$ time. The edge label and leaf label can be computed in $O(\log n)$ time. Inserting or deleting of a text T in C can be done in $O(|T| \log n)$ time.

How to construct a suffix tree efficiently is the critical problem of using suffix tree. Since *Weiner* proposed this data structure on 1973, a lot of work has been done in this area. The serial suffix tree construction is very mature and there are several well known algorithms which have linear complexity with respect to the length of the given string on both time and space. These algorithms include *Weiner*'s algorithm (1973), *McCreight*'s(1976) algorithm McC and *Ukkonen*'s algorithm (1992). Weiner's method is a monument on the character processing domain. McC uses suffix link technique to reduce the time complexity further and uses less space than the other two algorithms. The Ukkonen algorithm also uses the suffix link, and is an online arithmetic that builds suffix tree from left to right, i.e. adding t_{i+1} (a new character or label edge) to the current suffix tree $STree(T_n)$ and forms another new suffix tree $STree(T_{n+1})$, $0 < n \leq |T|$, where $|T|$ is the length of the string. In this study we use the *Ukkonen* algorithm.

3.4 Lexicon Construction

Our algorithm uses the known words (dictionary) generated from the previous stage (training processing) to segment the test corpus through the FMM heuristic rule. In order to improve the efficiency of matching, we can either sort all words in the dictionary according to the frequency of words or first classify words according to their length and then sort by their frequencies. However, these methods cannot solve the problem fundamentally. Our system stores the dictionary using a **Trie** structure, and sorts the items according to the order of the Chinese characters based on their Pinyin Romanization.

The construction steps are as follows:

1. The entire dictionary is stored as a Forest;
2. Each tree contains all the words which have the same first character;
3. All the second characters of these words in the same tree are children of the root node;
4. Other characters follow the same token.

For example, the first character of all the following words have the same Pinyin "bao": "爆炸", "爆炸装置", "爆炸药", "爆炸性", "爆发性质", "爆发性", "爆裂", "爆裂声", "爆裂声响", "爆破", "爆破突击队", "爆破井", "暴戾".

3.5 Comparison with Existing Methods

Previous research has used mutual information for Chinese word segmentation. For example, both *Chien et al.* (1997) and *Ong et al.* (1999), utilize MI in their key phrase extraction. Our proposed algorithm is different from these existing studies. First, MI is used in different ways and different stages in our segmenter. Chien et al. (1997) first split a given string into tokens with different lengths and use MI to filter out the strings with an MI value lower than the threshold. Ong et al. (1999) extend Chien's work by suggesting an updateable PAT-tree that allows the update of string frequencies dynamically. Different from their methods, we first split a given string coarsely, then compute MI of the neighbor tokens, and compare the MI value with the threshold. If the MI value is higher, then we merge the tokens and add the new word into the dictionary. Otherwise, we keep them unmerged. Another major difference is that we use a hybrid approach. In the first stage, we perform coarse splitting using a dictionary-based method to split the given texts, while the other two methods directly compound the characters according to their compositions.

4 Experiments

In order to evaluate the performance of our IASeg system, we compare it with the Hylanda segmenter (www.hylanda.com) and the ICTCLAS segmenter (Zhang et al., 2003). The Hylanda segmenter is a dictionary-based segmenter that has been widely used in practice (e.g., the search engine Zhongsou). ICTCLAS is an HMM-based segmenter. Both segmenters were chosen because they have shown very good performance in previous studies.

We use precision, recall, and F-measure to evaluate the performance of the segmenters. The calculations are as follows:

$$precision = \frac{correctNum}{autoTotalNum} \qquad recall = \frac{correctNum}{manualTotalNum}$$

$$F-measure = \frac{2 \times recall \times precision}{recall + precision}$$

where *correctNum* is the number of words correctly identified by the automatic method, *autoTotalNum* is the total number of words identified by the automatic method, and *manualTotalNum* is the number of words identified in the manual segmentation. A perfect segmenter will have a recall and precision of 100%. All these measures can be calculated automatically from a machine-segmented text, along with the human-segmented gold standard.

We collected a set of terrorism-related documents from the Web using crawlers. Within our list of *seed Websites* for terrorism-related content, we crawled news content using our page filter, which discards irrelevant materials like the advertising anchor text and the fringe links, and just keep the news content. As a result, we obtained a set of 330 news articles.

With these data, we setup our own gold set and training set, and run the three segmenters on the corpus. In our algorithm, the dictionary trie is updated dynamically after the first run of the training corpus. At last we obtained 57,339 words through the terrorism corpus, which also includes some high frequency general words.

The segmentation results of the three segmenters are shown in Table 2:

Table 2. Results of the three segmenters on terrorism-related content

	Hylanda	ICT	IASeg
Precision	0.8603	0.7759	0.9477
Recall	0.9160	0.8658	0.9480
F-measure	0.8874	0.8223	0.9477

Overall we can see that our segmenter achieves the best performance among the three segmenters in terms of precision, recall, and F-measure. Note that some of the ICT scores are *zero* because those tests could not be executed successfully due to, for example, problems with common Web page patterns like "……" or email addresses.

5 Discussion

Based on our testing of the segmenter on the terrorism-related corpus and other corpuses (not reported here), we found that two aspects of the training data have a profound influence on the model's accuracy. First, some errors are obviously caused by deficiencies in the training data, such as improperly segmented common words and name entities. Second, some errors stem from the topics covered by the corpus. It is not surprising that the error rate increases when the training and testing text represent different topic areas--such as training on military affairs news text and testing on medical text.

We observe that our algorithm has the following characteristics:

1. Using Mutual Information value as the new words identification threshold is greatly different from simple term frequency confidence.
2. Different thresholds will achieve different results. For example, a threshold of *20* may just keep all the tokens in their original form, while a threshold of *9* will result in merging some high co-occurrence adjacent tokens as one word. In general, we found that a lower threshold will make the segmenter to prefer longer words, thus resembling more closely with named entity extraction tools.
3. By using suffix tree, we can do searching and matching more easily and efficiently. Using the *Ukkonen* algorithm, we can construct the suffix tree in *O(n)* time complexity and *O(n)* space complexity.
4. Through our improved bigram structure, we can filter the low MI token-pairs, which greatly improves the boundary forecast accuracy.

6 Conclusion

In this paper, we propose a method on Chinese word segmentation based on suffix tree and mutual information. We integrate character-level information and word-level information and achieve encouraging results in segmenting a terrorism-related corpus. Our algorithm uses a two-stage statistical word segmentation. In the first stage, word suffix tree are used to generate a dynamic dictionary and N-gram model on input text, and then a hybrid approach is employed in the second stage to incorporate word N-gram probabilities, and mutual information with word-formation patterns to detect Out-Of-Vocabulary words.

Our future work includes the following:

- Improve our strategies by adding more words' position information and part-of-speech to develop an integrated segmenter which can perform known word segmentation and unknown word identification at the same time.
- Address the OAS (overlap ambiguity string) problem using syntax rules and address the "CAS" (combination ambiguity string) problem using SVM classifier.
- Study the possibility of performing Chinese named entity recognition using the HMM-based tagger and its integration with this Chinese analyzer.
- Investigate the problem of event information extraction based on syntax structure.

Acknowledgments

This work is supported in part by NNSFC #60621001 and #60573078, MOST #2006CB705500, #2004CB318103, and #2006AA010106, CAS #2F05N01 and #2F07C01, and NSF #IIS-0527563 and #IIS-0428241. We thank our team member Qingyang Xu for his help with the experiments. We also thank Fenglin Li and Shufang Tang for their help with data preparation and processing.

References

1. Chan, H.L., Hon, W.K., Lam, T.W., Sadakane, K.: Dynamic dictionary matching and compressed suffix trees. Society for Industrial and Applied Mathematics, 13–22 (2005) ISBN:0-89871-585-7
2. Chau, M., Xu, J.: Mining Communities and Their Relationships in Blogs: A Study of Online Hate Groups. International Journal of Human-Computer Studies 65(1), 57–70 (2007)
3. Chen, H., Xu, J.: Intelligence and Security Informatics. Annual Review of Information Science and Technology 40, 229–289 (2006)
4. Chen, M.T., Seiferas, J.: Efficient and elegant subword-tree construction. In: Combinatorial Algorithm on Words, NATO Advanced Science Institutes. Series F, vol. 12, pp. 97–107. Springer, Berlin (1985)
5. Chien, L.F.: PAT-Tree Based Keyword Extraction for Chinese Information Retrieval. In: ACM SIGIR (1997)
6. Creutz, M., Lagus, K.: Unsupervised Models for Morpheme Segmentation and Morphology Learning. ACM Transactions on Speech and Language Processing 4(1) (January 2007)
7. Cui, S.Q., Liu, Q., Meng, Y., Yu, H.: Nishino Fumihito. New Word Detection Based on Large-Scale Corpus 43(05), 927–932 (2006)

8. Dai, Y.B., Khoo, S.G.T., Loh, T.E.: A new statistical formula for Chinese word segmentation incorporating contextual information. In: Proc. of the 22nd ACM SIGIR Conf. on Research and Development in Information Retrieval, pp. 82–89 (1999)
9. Fang, Y., Yang, H.E.H.: The Algorithm Design and Realization to Calculate The Mutual Information of Four- Word- String in Large Scale Corpus. Computer Development & Applications 1 (2005)
10. Giegerich, R., Kurtz, S.: From Ukkonen to McCreight and Weiner: A unifying view to linear-time suffix tree construction. Algorithmica 19, 331–353 (1997)
11. Hockenmaier, J., Brew, C.: Error-driven segmentation of Chinese. Communications of COLIPS 1(1), 69–84 (1998)
12. Jia, N., Zhang, Q.: Identification of Chinese Names Based on Maximum Entropy Model. Computer Engineering 33(9), 31–33 (2007)
13. Li, J.F., Zhang, Y.F.: Segmenting Chinese by EM Algorithm. Journal of the China Society for Scientific and Technical Information 03, 13–16 (2002)
14. Li, R., Liu, S.H., Ye, S.W., et al.: A method of crossing ambiguities in Chinese word segmentation based on SVM and k-NN. Journal of Chinese Information Processing 15(6), 13–18 (2001) (in Chinese)
15. Maaß, M.: Suffix Trees and their Applications. Ferienakademie 1999 Kurs 2: Bäume-Algorithmik and Kombinatorik (1999)
16. Low, J.K., Ng, H.T., Guo, W.: A Maximum Entropy Approach to Chinese Word Segmentation. In: Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing, Jeju Island, Korea, pp. 161–164 (2005)
17. McCreight, E.M.: A space-economical suffix tree construction algorithm. Journal of ACM 23(2), 262–272 (1976)
18. Ong, T.H., Chen, H.: Updateable PAT-Tree Approach to Chinese Key Phrase Extraction using Mutual Information: A Linguistic Foundation for Knowledge Management. In: Proceedings Asian Digital Library Conference, Taipei, Taiwan, pp. 63–84 (1999)
19. Palmer, D.: A trainable rule-based algorithm to word segmentation. In: Proceedings of the 35th Annual Meeting of the Association of Computational Linguistics, Madrid, Spain (1997)
20. Peng, F.C., Feng, F.F., McCallum, A.: Chinese segmentation and new word detection using conditional random fields. In: COLING 2004, Geneva, Switzerland (2004)
21. Peng, F.C., Dale, S.: Self-supervised Chinese Word Segmentation. In: Proceedings of the 4th International Symposium of Intelligent Data Analysis, pp. 238–247 (2001)
22. Ponte, J.M., Croft, W.B.: Useg: A retargetable word segmentation procedure for information retrieval. In: Proceedings of SDAIR 1996, Las Vegas, Nevada (1996)
23. Rabiner, L.: A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE 77(2), 257–286 (1989)
24. Sproat, R., Shih, C., Gale, W., Chang, N.: A stochastic finite-state word-segmentation algorithm for Chinese. Computational Linguistics 22(3), 377–404 (1996)
25. Sun, M.S., Xiao, M., Zou, J.Y.: Chinese Word Segmentation without Using Dictionary Based on Unsupervised Learning Strategy. Chinese Journal of Computers 27(6), 736–742 (2004)
26. Teahan, W.J., Wen, Y., McNab, R.J., Witten, I.H.: A compression-based algorithm for Chinese word segmentation. Computational Linguistics 26, 375–393 (2000)
27. Ukkonen, E.: Constructing Suffix Trees On-Line in Linear Time. In: Leeuwen, J.v. (ed.) Algorithms, Software, Architecture, Proc. IFIP 12th World Computer Congress, Information Processing 1992, Madrid, Spain, vol. 1, pp. 484–492 (1992)
28. Ukkonen, E.: On-line Construction of Suffix-Trees. Algorithmica 14(3) (1995)

29. Weiner, P.: Linear Pattern Matching Algorithms. In: Proc. 14th IEEE Annual Symp. on Switching and Automata Theory, pp. 1–11 (1973)
30. Wu, Z., Tseng, G.: Chinese text segmentation for text retrieval achievements and problems. JASIS 44(9), 532–542 (1993)
31. Xue, N.W., Chiou, F.-D., Palmer, M.: Building a large annotated Chinese corpus. In: The Proceedings of the 19th International Conference on Computational Linguistics, Taipei, Taiwan (2002)
32. Xue, N.W.: Chinese word segmentation as character tagging. International Journal of Computational Linguistics and Chinese Language Processing 8(1), 29–48 (2003)
33. Yu, H.K., Zhang, H.P., Liu, Q., Lv, X.Q., Shi, S.C.: Chinese named entity identification using cascaded hidden Markov model. Journal on Communications 27(2), 87–94 (2006)
34. Zhang, H.P., Yu, H.K., Xiong, D.Y., Liu, Q.: HMM-Based Chinese lexical analyzer ICTCLAS. In: Proc. of the 2nd SIGHAN Workshop, pp. 184–187 (2003)
35. Zhang, Ch.L., Hao, F.L., Wan, W.L.: An automatic and dictionary-free Chinese word segmentation method based on suffix array. Journal of Jilin University (Science Edition) 4 (2004)
36. Zhou, L.X., Liu, Q.: A Character-net Based Chinese Text Segmentation Method. In: SEMANET: Building and Using Semantic Networks Workshop, attached with the 19th COLING, pp. 101–106 (2002)