Part-of-Speech Tagging for Old Chinese

Liang Huang[1], Yi-Nan Peng[1], and Huan Wang[2]

[1] Department of Computer Science, Shanghai Jiaotong University
No. 1954 Huashan Road, Shanghai
P.R. China 200030

[2] Department of Chinese Literature and Linguistics, East China Normal University
No. 3663 North Zhongshan Road, Shanghai,
P.R. China 200062

Abstract Old Chinese is essentially different from Modern Chinese, in both grammar and morphology. While there has recently been a great deal of work on part-of-speech (POS) tagging for modern Chinese, the POS of Old Chinese is largely neglected. To the best of our knowledge, this is the first work in this area. Fortunately however, in terms of tagging, Old Chinese is easier than modern Chinese in that most Old Chinese words are single-character-formed, requiring no segmentation. So in this paper, we will propose and analyze a simple statistical approach for POS tagging of Old Chinese. We first designed a tagset for Old Chinese that is later shown to be accurate and efficient. Then we apply the hidden markov model (HMM) together with the Viterbi algorithm and made several improvements, such as sparse data problem handling, and unknown word guessing, both designed especially for Chinese. As the training set grows larger, the hit rate for bigram and trigram increases to 94.9% and 97.6%, respectively. The importance of our work lies in the previously unseen features that are special for Old Chinese and we have developed successful techniques to deal with them. Although Old Chinese is now a dead language, this work still has many applications in such areas as Ancient-Modern Chinese Machine Translation.

Key Words. Part-of-speech tagging, Old Chinese, hidden markov model, Viterbi algorithm, bigram, trigram, statistical approach

1 Introduction

Part-of-speech tagging is fundamental in natural language processing. It selects the most likely sequence of syntactic categories (part-of-speech) for the words in a sentence, and passes its output to the next processing level, usually a syntactic parser. Over the last twenty years, the correctness of POS tagging has increased dramatically on some famous English corpora like the Penn Treebank Project [4]. And the POS tagging for Chinese has also resulted in very high hit rates [8]. There are many machine learning approaches for automated POS tagging, and the most successful of them are rule-based methods and statistical methods. So in this section, we will first briefly summarize the different approaches of POS tagging, then point out the particularities of Chinese and Old Chinese POS, and finally gives the organization of the rest of the paper.

1.1 Rule-Based Approach

Typical rule based approaches [9] use contextual information to assign tags to ambiguous words. Generally rule-based taggers have error rates substantially higher than the state-of-the-art statistical taggers. In [4], a highly competitive tagger is described which captures the learned knowledge in a set of simple deterministic rules instead of a large table of statistics. And in [5], a transformation-based error-driven tagger was developed with contextual rules as *patches*, which greatly increases the hit rates.

1.2 Statistical Approach

Stochastic taggers [2, 3, 6, 8, 10] have obtained a high degree of accuracy without performing any syntactic analysis of the input. There are many methods inside the statistical model, among them are the famous hidden markov model [2, 3, 8, 10] and the maximum entropy approach [6].

The hidden markov model is the most widely used model for POS tagging. It originates from the Viterbi algorithm [1]. In this model, word-tag probabilities and n-gram probabilities (parameters in the hidden markov model) are obtained from the training set, usually a manually annotated corpus.

The maximum entropy approach offers a better use of contextual information. The experiment on Penn Treebank Wall St. Journal shows that this method gives a high hit rate than previous HMM models [6].

1.3 The Limitations and Improvements of HMM

The main limitations of HMM are sparse data problem (especially for trigram) and unknown word guessing. A number of methods have been developed to solve these two problems [10, 13, 4, 5].

Sparse Data Problem. We have to move from bigrams to tri-grams, 4-grams and more to include more tags into the consideration. Then the problem of the *sparse data set* or inadequate parameters emerges [8]. In [10], a *smoothing* method is developed to solve it using interpolations of uni-, bi- and tri-grams. But this approach requires too much computation at the training phase.

Unknown Word Handling. Previous works show that the hit rates for unknown words are substantially lower than those known words [13, 10]. In [4] and [5], guessing from suffixes is proposed, but it is not applicable to Chinese. And in [13], a *supporting vector* approach is presented, yet still too time-consuming.

So in this paper, we will present simple-yet-effective methods to handle these 2 problems.

1.4 POS Tagging of Chinese and Old Chinese

Most of the taggers were designed for English, with a little for other European languages. Chinese, however, is totally different from the Indo-European languages in its special grammar. And some features discussed above cannot be transplanted to the tagging of Chinese directly.

Word Segmentation. Modern Chinese is written without spaces to separate between words. For example, consider the following phrase

江 泽 民 主 席
jiang ze min zhu xi

can be segmented as

江泽民 / 主席
jiang ze min / zhu xi
(in English, President Jiang Ze-Ming)

and can be also segmented in a different way

江泽 / 民主 / 席
jiang ze / min zhu / xi
(In English, rivers and ponds / democracy / seat)

Apparently, the second segmentation is nonsense. So successful segmentation is the first step of POS tagging [8], which made tagging for Chinese much more difficult than taggers for European languages.

Generally speaking, Old Chinese is even more difficult than modern Chinese, in its obsolete grammar patterns. But actually, from the tagger's point of view, Old Chinese is somewhat easier, because most words are written in the single-character form, thus requiring no need of word segmentation.

Punctuation. Unfortunately, Old Chinese is written without any punctuations, and all the inputs to our program are manually punctuated. This is another specialty of Chinese processing. Fortunately most Old Chinese documents have already been manually punctuated in the 20th century, so our work is still right-to-use for applications.

Unknown word guessing. Another area where methods for European languages are not applicable to Chinese Processing. In most publications for tagging European languages, a common approach using suffixes or surrounding context of unknown words is used [10, 5]. But neither suffixes nor capitalization is extant in Chinese. Especially in Old Chinese, a character is a word, so no separation of word is possible. Because in computer, Chinese is presented by GB/BIG5 code and we cannot get any information of meaning or structure from the code. So we will present our own approach for handling unknown words.

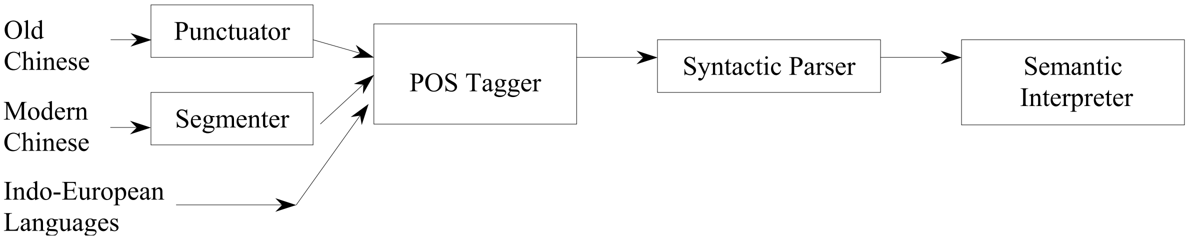

Fig. 1. The complete processing procedure of Modern/Old Chinese by comparison to the Indo-European Languages

The rest of the paper is organized as follows. In section 2, we give a summary of lexical analysis of Old Chinese and discuss the design the tagset. The dynamic-programming-based tagging algorithm and features are presented in Section 3. Algorithm performance is evaluated in Section 4. In Section 5, we give a conclusion of the paper and briefly explore the applications and future work.

2 Old Chinese Corpus and Tagset

The design of tagset is very crucial to the accuracy and efficiency of the tagging algorithm, and this was largely neglected in the literature where almost all the researchers use those famous corpora and their tagset as the standard test-beds. In addition, the ambiguity of Old Chinese is substantially higher than Indo-European languages. So in this section, we will focus on the construction of corpus and design of the tagset.

2.1 The Corpus

The only established corpus of Old Chinese is the Taiwan-Corpus [12]. But for the purpose of evaluating our own tagging algorithm and tagset, we constructed a small-sized corpus from the classical Old Chinese documents like *论语* (lun yu, by the students of Confucius) and *道德经* (dao de jin, by Laozi, the father of Daoism) and some other classics.

Our selection criteria of texts is: omit those with proper nouns, omit those with very hard words, and omit grammatical exceptions. As these three assumptions actually holds for most Old Chinese Documents, our corpus does reflect typical Old Chinese. And finally we got a corpus of about 1000 words as the training set. For the test set, the three criteria still holds. And we selected a relatively simple text from *荀子* (by Xunzi, another great figure of Confucianism). The length of the test set is about 200 words.

2.2 Tagset

There is a tradeoff in the design of the tagset. The sizes of tagsets in previous works vary from smaller than 20 to larger than 400. On one hand, in order to obtain a high accuracy of automatic tagging, we divide the basic lexical categories like verb and adjective into sub-categories, like adjective used as attributive or as predicate. Obviously these discriminations are crucial, but it does severe the sparse data set problem. Considering the small corpus in this work, we did not use such *accurate* tagsets as in [8].

As Chinese grammar focus on the sequence of words rather than the morphological information, it is much more ambiguous than languages with inflexions like Indo-European languages. In other words, contextual information contributes more to the part-of-speech tagging than lexical information. Taking this into account, we designed a tagset with special interest not only to the lexical categories, but also the categories of components a word may belong. For example, we discriminate adjectives into 4 subcategories like *Adjective as attributive*, etc. (See table 1). This discrimination turns out to be an important contributing factor of the tagging accuracy. (See Section 4).

Note that we map punctuations into 2 sets: *period* and *comma*.

Table 1. Tagset for Old Chinese

No.	**Tag Name**	**Meaning**	**English meaning**
0	N	名词	Noun
1	Aa	形容词作定语	Adjective as attributive
2	Aw	形容词作谓语	Adjective as verbal phrase
3	Az	形容词作状语	Adjective as adverbial
4	Ab	形容词作表语	Adjective as predicate

5	Ad	副词	Adverb
6	Vi	不跟宾语的动词	Verb without object
7	Vt	跟宾语的动词	Verb with object
8	Vy	意动	*Special for Old Chinese*
9	Vs	使动	*Special for Old Chinese*
10	Vb	省略宾语的动词	Verb with object omitted
11	Vx	系动词	Predicate verb
12	Vyou	动词“有”	The Verb *have*
13	Vyue	动词“曰”	The Verb *say*
14	Conj	连词	Conjunction
15	Yq	语气词	Exclamation
16	Prep	带宾语的介词	Preposition with object
17	Prepb	省略宾语的介词	Preposition with object omitted
18	Num	数词	Number
19	Qpron	疑问代词	Wh-pronoun
20	Npron	名词性代词	Noun-pronoun
21	Apron	形容词性代词	Adjective-pronoun
22	Za	“之”作定语后置标志	*Special for Old Chinese*
23	Zj	“者”作名词性词尾	*Special for Old Chinese*
24	Zd	“之”作“的”	*Special for Old Chinese*
25	Zw	“之”作取消主谓独立性标志	*Special for Old Chinese*
26	Fy	发语词	*Special for Old Chinese*
27	Period	终止性标点	。；？！
28	Comma	停顿性标点	，、：

3 Tagging Algorithms and Features

Our tagging algorithm is based on the Hidden Markov Model and has several improvements.

Let w_1, ...w_T be a sequence of words. We want to find the sequence of POS tags t_1, ...t_T that maximizes the probability

$$\Pr(t_1,...,t_T \mid w_1,...,w_T) \quad \textbf{(1)}$$

and according to Bayes' rule [11], it equals

$$(\Pr(t_1,...,t_T)\cdot\Pr(w_1,...,w_T \mid t_1,...,t_T)) / \Pr(w_1,...,w_T) \quad \textbf{(2)}$$

as the denominator is pre-determined, we only need to find the sequence t_1, ...t_T that maximizes the formula

$$\Pr(t_1,...,t_T)\cdot\Pr(w_1,...,w_T \mid t_1,...,t_T) \quad \textbf{(3)}$$

Obviously, the above formula requires far too much data to calculate accurately. So we use the n-gram hidden markov models.

3.1 Hidden Markov Model

As stated before, POS taggers seldom use n-grams for n>3, due to the *sparse data problem*. So in this paper, we use uni-grams, bi-grams and tri-grams.

Different from traditional HMM, we denote lexical frequencies to be *word-tag* probabilities, not *tag-word* probabilities. Though in this way it doesn't conform to the Bayes' rule, it is even more effective in POS tagging of Chinese, as stated in [8], and we also found it much easier when handling unknown words.

We denote unigrams, bigrams, and trigrams probabilities as follows,

$$\textit{Unigrams} \qquad \Pr(t) = \frac{f(t)}{N} \tag{4}$$

$$\textit{Bigrams} \qquad \Pr(t_1 \mid t_2) = \frac{f(t_1, t_2)}{f(t_2)} \tag{5}$$

$$\textit{Trigrams} \qquad \Pr(t_1 \mid t_2, t_3) = \frac{f(t_1, t_2, t_3)}{f(t_2, t_3)} \tag{6}$$

$$\textit{Lexical} \qquad \Pr(t \mid w) = \frac{f(t, w)}{f(w)} \tag{7}$$

for all t_1, t_2, t_3 in the tagset and w in the lexicon. N is the total number of tokens in the training set.

Now for bigram model, we need to maximize the following:

$$P = \Pi \Pr(t_i \mid t_{i-1}) \cdot \Pr(t_i \mid w_i) \tag{8}$$

For trigram model, we need to maximize the following:

$$P = \Pi \Pr(t_i \mid t_{i-2}, t_{i-1}) \cdot \Pr(t_i \mid w_i) \tag{9}$$

3.2 Dynamic Programming Algorithm

Our tagging algorithm is based on the Viterbi Algorithm [1,11], which is Dynamic Programming in nature.

For the simplicity of programming and accuracy of tagging, we intentionally add a *period* tag before each sentence and assume that each sentence ends with a *period.* This method is better than the traditional *loose end* in other publications [10].

Bigram model

For bigram model, the dynamic programming algorithm is as follows (in pseudo-code)

```
list[0]="period";
for i=1 to len
  for j=0 to tagnum-1
    best[i][j]=$\text{MAX}_{k=0\sim tagnum-1}$ (
      best[i-1][k]*prob[list[i]]*markov[k][j])
    pre[i][j]=index of k that gave the max above
```

Trigram model

For trigram model, the algorithm needs some considerations. For the first word, as it is just after the first intentional *period,* it has fewer than 2 precedings; and for words just after any punctuation in the sentence, the same problem also occurs. So here we use bigrams results for these occasions. The trigram algorithm is as follows

```
list[0]="period";
for i=1 to len
  if list[i-1] is a punctuation then
            use bigrams results instead
  else
```

```
for j=0 to tagnum-1
  for l=0 to tagnum-1
    best3[i][l][j]=$\mathrm{MAX}_{k=0\sim tagnum-1}$ (
                best3[i-1][k][l] *markov3[k][l][j]
                  *prob[list[I]][j])
    pre3[i][l][j]=index of k that gave the max above
```

3.3 Unknown Word Handling

Recall that most methods used in unknown words guessing for European languages are not applicable to Chinese or Old Chinese. So in this paper we present a simple but novel approach.

For any word that does not occur in the training set, we denote its word-tag probabilities to be the unigrams probabilities for each tag. For example, for an unknown word *w,* we have

$$\Pr(t_i \mid w) = unigram(t_i) = \frac{f(t_i)}{N} \tag{10}$$

N is the total number of words in the training set.

Though it violates the classical probability theory, experiments (see Section 4) indicate it quite effective, especially for trigram model. The underlying reason for its success is the independence of *phases* in Dynamic Programming.

3.4 Sparse Data Problem

Besides the unknown word handling, the problem of *sparse data set* is also difficult to handle. As stated before, the size of our corpus is very small, compared to well-established English corpora. So we need a technique to unknown sequences, i.e. unknown bigram pairs or trigram sequences. Rather than the complicated *smoothing* method in [10], we here present a simpler approach made of 2 steps:

1. Add a very small number, usually 10^{-60}, to all lexical probabilities. Obviously it will not affect the correct probabilities learned from the corpus. But by this addition, we enable the algorithm for guessing the most probable context sequence, when word-tag pairs sequence is not presented in the corpus. This is effective especially when context information is quite deterministic (provides enough hints) at certain positions.
2. In the dynamic programming, if still no possible sequence at certain position, use the lower-level n-gram, i.e., for bigram model, use lexical probabilities, and for trigram, use big ram's results. It is an expedient approach, yet still works quite well as experiments indicated.

4 Performance Evaluations

4.1 Accuracy

We made the learning set enlarging slowly and test the performance (learning curves) of the tagset and the algorithm. As the training set grows larger, the result is show in the following figures. We not only test the overall hit rates, but also the known and unknown hit rates separately.

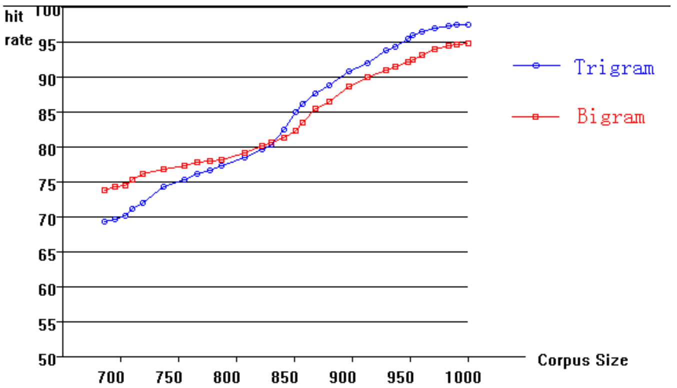

Fig. 2. Learning Curves for the overall hit rate: *bigram model* vs. *trigrams model.* The learning length increases from 686 to 1000 tokens.

Figure 2 shows the learning curves of the taggers, i.e., the accuracy depending on the amount of training data. Corpus size is the training length, namely, the number of tokens used for training. As corpus expanding, the hit rate of bigrams and trigrams increases from 73.9% to 94.9% and from 69.5% to 97.6%, respectively. At the beginning, when corpus is small, trigram hit rates are lower than bigram hit rate mainly because of the sparse data problem. And at the end, when contextual information is more abundant, trigrams overrun bigrams substantially.

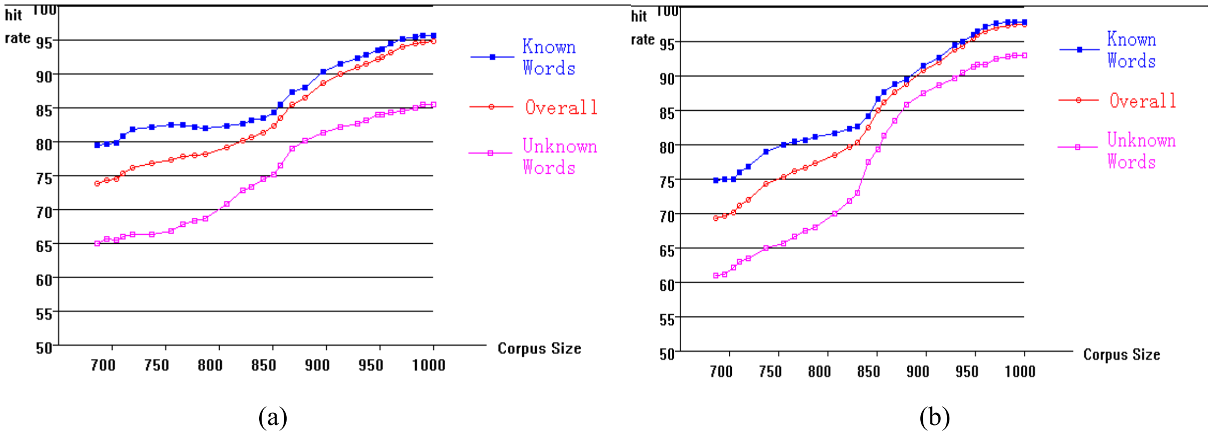

Fig. 3. Learning Curves of known words and unknown words. (a) Bigram model. (b) Trigram model. The percentage of unknown words decreases from 39.4% at the beginning to 8% at the end, almost linearly.

Figure 3 shows the learning curves of known words and unknown words. Different from most POS tagging works found in the literature where the accuracies for known words are very high even for very small corpora [11, 10], our work shows that for tagging of Old Chinese, the beginning accuracy for known words is rather low. In [10], the author concludes that even considering the lexical frequencies only, the hit rate would be about 90 percent, primarily because over half of the words appearing in most corpora are *not* ambiguous. But for Old Chinese, a language full of lexical ambiguity, with an accurate tagset used, the situation is entirely different. Figure 3 shows the beginning accuracy for known words are only 79.7% and 74.9% for bigrams and trigrams, respectively.

Unknown word guessing is another successful part of our work. Figure 3 shows our guessing technique very effective. The beginning hit rates for bigrams and trigrams are 65.0% and 60.9%, respectively, about the same with most successful works in the literature [6, 10, 13]. But the learning curve for unknown words rises rapidly and at the end, the hit rate are 85.1% and 93.2%, a little higher than those previous works. The high hit rate of unknown words guessing of trigram model shows that our technique is especially effective when contextual information (hint) is quite deterministic at certain position.

4.2 Error Analysis

Though the overall hit rates are high, there are still many errors made by the taggers. Obviously, bigrams made much more errors than trigrams, when corpus is relatively large. The typical errors made by bigrams are those grammatical structures special in Old Chinese that *must* be recognized from at least 3 words, regardless of the training length. The following is a list of bigram errors.

1. the *pre-positioned object structure*, a special structure called 宾语前置 (bin yu qian zhi) in Old Chinese.

bigram 古n 之zd 人n 不ad 余npron 欺vi。
correct 古n 之zd 人n 不ad 余npron 欺vt。

2. *prepositional phrase as adverbial:* *bigram* 不ad 为vt 尧n 存vi,
correct 不ad 为*prep* 尧*n* 存vi,

3. *Vt+n* or *ad+vi:* *bigram* 人*n* 而*conj* 无*ad* 信*vi*，
correct 人*n* 而*conj* 无*vt* 信*n*，

The above errors have all been corrected in trigram model. And there are very few trigram errors. Typical errors are those with unknown words.

5 Conclusion and Future Work

In this paper we have proposed and analyzed a simple corpus-based statistical method of part-of-speech tagging for Old Chinese texts. To the best of our knowledge, this is the first work in the area of POS tagging of Old Chinese.

A special tagset is first designed for Old Chinese. We then base our work on the hidden markov model (HMM) model and the Viterbi algorithm. In addition, several features, such as sparse data problem handling are presented and we also developed an unknown word guessing schema especially for Chinese. We constructed a small-size corpus of Old Chinese classics and selected a typical and simple text as the test set. As the training set grows larger, the hit rate increases to 94.9% for bi-gram and 97.6% for tri-gram. The most important innovation of our work is that we have presented many previously unseen features that are special for Old Chinese and we have developed successful techniques to deal with them.

For applications, although Old Chinese is now a dead language, our work is still useful in mainly two areas: Ancient-Modern Chinese Machine Translation and Information Retrieval of Old Chinese. Old Chinese is known to be most abundant source of information for ancient and medieval civilization, when China is the leading country of the world. And the percentage of Old Chinese documents that have been translated to modern Chinese is very low. So the above two areas have a very bright future.

For future work, as stated before, the computational study of Old Chinese is just commencing. And our future work includes a syntactic chart-based bottom-up parser, for a probabilistic context-free grammar (PCFG) of Old Chinese. With the forward-backward algorithm which provides context-dependent information, the chart-based parser can be statistically accurate.

Another possible work is the sentence-punctuator. We think by a few changes in the HMM model will yield an effective punctuator.

Last but most important, we need larger corpora. The corpus used in our program is too small, and there are very few annotated Old Chinese corpora. So we may first construct a manually annotated medium-sized corpora for future study.

References

1. Viterbi, A.: Error bounds for convolution codes and an asymptotically optimal decoding algorithm. IEEE Trans. on Information Theory 13:260-269. 1967
2. Leech, G. et al.: The Automatic Grammatical Tagging of the LOB Corpus, ICAME News, 7 (1983), 13-33.
3. Merialdo, B.: Tagging Text with a Probabilistic Model, IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 1991. 809-812.
4. Brill, E.: A simple rule-based part-of-speech tagger, Proceeding of the 3rd Conference on Applied Natural Language Processing(ACL), 1992, pp 152-155.
5. Brill, E.: Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging. Computational Linguistics, 21(4), 1995 pages 543–565.
6. Ratnaparkhi,A. et al.: A Maximum Entropy Model for Part-of-Speech Tagging. In Proceedings of Conference on Empirical Methods in Natural Language Processing(EMNLP-1), 1996, pages 133–142
7. Charniak, E. et al. : Equations for Part-of-Speech Tagging. In Proceedings of the Eleventh National Conference on Artificial Intelligence(AAAI-93), 1993. pages 784–789.
8. Lua, K.: Part of Speech Tagging of Chinese Sentences Using Genetic Algorithm, Proceedings of Conference on Chinese Computing, Singapore, Jun. 1996, pp. 45-49
9. Hindle, D. Acquiring disambiguation rules from text. In Proceedings of 27th Annual Meeting of the Association for Computational Linguistics, 1989
10. Brant, T.: TnT — A Statistical Part-of-Speech Tagger. In Proceedings of the 6th Applied NLP Conference(ANLP-2000), 2000, pages 224–231.
11. Allen, J.: Natural Language Understanding, The Benjamin/Cummings Publishing Company, Inc. 1995
12. Wei, P. et al. : Historical Corpora for Synchronic and Diachronic Linguistics Studies, Pacific Neighborhood Consortium, 1997
13. Nakagawa, T. et al. : Unknown Word Guessing and Part-of-Speech Tagging Using Support Vector Machines, Proceedings of the 6th Natural Language Processing Pacific Rim Symposium, 2001