A Chinese Efficient Analyser Integrating Word Segmentation, Part-Of-Speech Tagging, Partial Parsing and Full Parsing

GuoDong ZHOU
Institute for Infocomm Research
21 Heng Mui Keng Terrace
Singapore, 119613
zhougd@i2r.a-star.edu.sg

Jian SU
Institute for Infocomm Research
21 Heng Mui Keng Terrace
Singapore, 119613
sujian@ i2r.a-star.edu.sg

Abstract

This paper introduces an efficient analyser for the Chinese language, which efficiently and effectively integrates word segmentation, part-of-speech tagging, partial parsing and full parsing. The Chinese efficient analyser is based on a Hidden Markov Model (HMM) and an HMM-based tagger. That is, all the components are based on the same HMM-based tagging engine. One advantage of using the same single engine is that it largely decreases the code size and makes the maintenance easy. Another advantage is that it is easy to optimise the code and thus improve the speed while speed plays a critical important role in many applications. Finally, the performances of all the components can benefit from the optimisation of existing algorithms and/or adoption of better algorithms to a single engine. Experiments show that all the components can achieve state-of-art performances with high efficiency for the Chinese language.

1 Introduction

Traditionally, a text parser outputs a complete parse tree for each input sentence, achieving a speed in the order of 10 words per second (wps) (Abney 1997). However, for many applications like text mining, a parse tree is not necessary and a speed of 10 wps is unacceptable when we have to process millions of words in thousands of documents in a reasonable time (Feldman 1997). Therefore, there is a compromise between speed and performance in many applications.

The objective of this paper is to develop an efficient analyser for the Chinese language, exploring different intermediate forms, achieving a target speed in the region of 1,000 wps for full parsing, 2,000 wps for partial parsing and 10,000 wps for word segmentation and part-of-speech tagging, with state-of-art performances.

The layout of this paper is as follows. Section 2 describes the Chinese efficient analyser. Section 3 presents the HMM and the HMM-based tagger. Sections 4 and 5 describe the applications of the HMM-based tagger in integrated word segmentation and part-of-speech tagging, partial parsing, and full parsing respectively. Section 6 gives the experimental results. Finally, some conclusions are drawn with possible extensions of future work in section 7.

2 Chinese Efficient Analyser

The Chinese efficient analyser can be described by the example as shown in Figure 1. Here, "." in Figure 1 means that the current node has not been chunked till now. For convenience, it is regarded as a "special chunk" in this paper and others as "normal chunks". Therefore, every node in Figure 1 can be represented as a 3tuple $c_i(p_i, w_i)$, where c_i is the i-th chunk in the input chunk sequence and

- w_i is the head word of c_i and p_i is the POS tag of w_i when $c_i \neq .$ (c_i is a normal chunk). In this case, we call node $c_i(p_i, w_i)$ a normal chunk node.

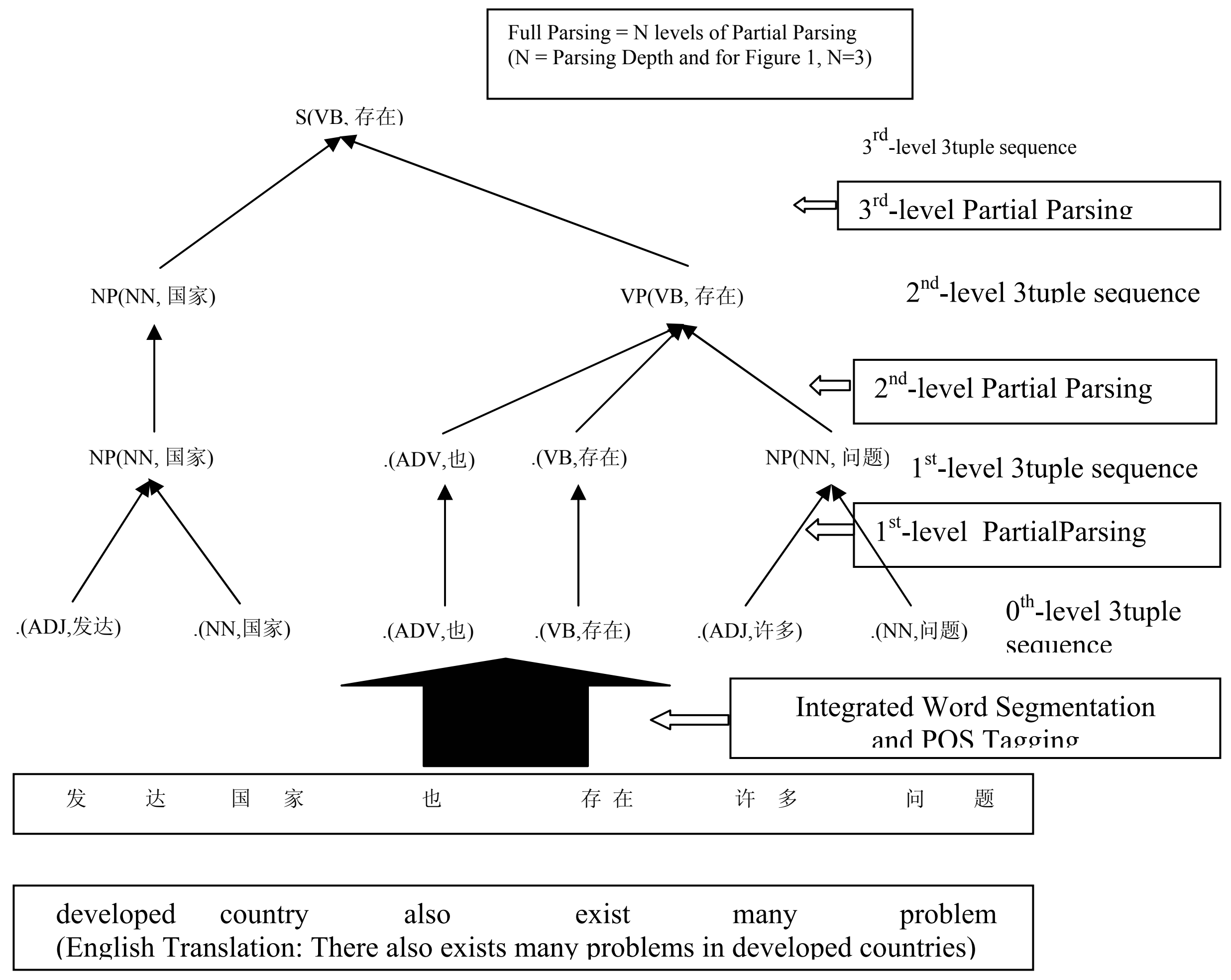
Full Parsing = N levels of Partial Parsing
(N = Parsing Depth and for Figure 1, N=3)
S(VB, 存在)
3rd-level 3tuple sequence
3rd-level Partial Parsing
NP(NN, 国家)
VP(VB, 存在)
2nd-level 3tuple sequence
2nd-level Partial Parsing
NP(NN, 国家)
.(ADV,也)
.(VB,存在)
NP(NN, 问题)
1st-level 3tuple sequence
1st-level PartialParsing
.(ADJ,发达)
.(NN,国家)
.(ADV,也)
.(VB,存在)
.(ADJ,许多)
.(NN,问题)
0th-level 3tuple sequence
Integrated Word Segmentation and POS Tagging
发 达 国 家 也 存 在 许 多 问 题
developed country also exist many problem
(English Translation: There also exists many problems in developed countries)

Figure 1: An Example Sentence "发达国家也存在许多问题" of the Chinese Efficient Analyser

- w_i is just the word linked with c_i and p_i is the POS tag of w_i when $c_i = .$ (c_i is a special chunk). In this case, we call node $c_i(p_i, w_i)$ a special chunk or POS node.

Figure 1 shows that, sequentially from bottom to top,

1) Given a Chinese sentence (e.g. "发达国家也存在许多问题"), it is segmented and tagged into a sequence of special chunk, POS and word 3tuples (.(ADJ, 发达) .(NN, 国家) .(ADV, 也) .(VB, 存在) .(ADJ, 许多) .(NN, 问题)) via the integrated word segmentation and POS tagging. In this paper, this resulting 3tuple sequence is called "0th-level 3tuple sequence".

2) The 0th-level 3tuple sequence is then chunked into 1st-level 3tuple sequence (NP(NN, 国家) .(ADV, 也) .(VB, 存在) NP(NN, 问题)) via 1st-level partial parsing, while POS nodes .(ADJ, 发达) and .(NN, 国家) are chunked into a normal chunk node NP(NN, 国家), and POS nodes .(ADJ, 许多) .(NN, 问题) into NP(NN, 问题).

3) The 1st-level 3tuple sequence is further chunked into 2nd-level 3tuple sequence (NP(NN, 国家) VP(VB, 存在)) via 2nd-level partial parsing, while mixed POS and normal chunk nodes .(ADV, 也) .(VB, 存在) NP(NN, 问题) are chunked into a normal chunk node VP(VB, 存在).

4) Finally, 2nd-level 3tuple sequence is chunked into 3rd-level 3tuple sequence (S(VB, 存在)) via 3rd-level partial parsing, while normal chunk nodes NP(NN, 国家) and VP(VB, 存在) are chunked into a normal chunk node S(VB, 存在).
5) In this way, full parsing is completed with a fully parsed tree after several levels (3 in the example of Figure 1) of cascaded partial parsing.

3 HMM-based Tagger

The Chinese efficient analyser is based on the HMM-based tagger described in Zhou et al 2000a. Given a token sequence $G_1^n = g_1 g_2 \cdots g_n$, the goal of tagging is to find a stochastic optimal tag sequence $T_1^n = t_1 t_2 \cdots t_n$ that maximizes

$$\log P(T_1^n \mid G_1^n) = \log P(T_1^n) + \log \frac{P(T_1^n, G_1^n)}{P(T_1^n) \cdot P(G_1^n)}$$

By assuming mutual information independence:

$$MI(T_1^n, G_1^n) = \sum_{i=1}^{n} MI(t_i, G_1^n) \text{ or}$$

$$\log \frac{P(T_1^n, G_1^n)}{P(T_1^n) \cdot P(G_1^n)} = \sum_{i=1}^{n} \log \frac{P(t_i, G_1^n)}{P(t_i) \cdot P(G_1^n)}$$

we have:

$$\log P(T_1^n \mid G_1^n) = \log P(T_1^n) - \sum_{i=1}^{n} \log P(t_i) + \sum_{i=1}^{n} \log P(t_i \mid G_1^n)$$

Both the first and second items correspond to the language model component of the tagger. We will not discuss these two items further in this paper since they are well studied in ngram modeling. This paper will focus on the third item $\sum_{i=1}^{n} \log P(t_i \mid G_1^n)$, which is the main difference between our tagger and other HMM-based taggers. Ideally, it can be estimated by using the forward-backward algorithm (Rabiner 1989) recursively for the first-order (Rabiner 1989) or second-order HMMs (Watson et al 1992). To simplify the complexity, several context dependent approximations on it will be attempted in this paper instead, as detailed in sections 3 and 4.

All of this modelling would be for naught were it not for the existence of an efficient algorithm for finding the optimal state sequence, thereby "decoding" the original sequence of tags. The stochastic optimal tag sequence can be found by maximizing the previous equation over all the possible tag sequences. This is implemented via the well-known Viterbi algorithm (Viterbi 1967) by using dynamic programming and an appropriate merging of multiple theories when they converge on a particular state. Since we are interested in recovering the tag state sequence, we pursue 16 theories at every given step of the algorithm.

4 Word Segmentation and POS Tagging

Traditionally, in Chinese Language Processing, word segmentation and POS tagging are implemented sequentially. That is, the input Chinese sentence is segmented into words first and then the segmented result (in the form of word lattice or N-best word sequences) is passed to POS tagging component. However, this processing strategy has following disadvantages:

- The word lexicons used in word segmentation and POS tagging may be different. This difference is difficult to overcome and largely drops the system accuracy although different optimal algorithms may be applied to word segmentation and POS tagging.
- With speed in consideration, the two-stage processing strategy is not efficient.

Therefore, we apply the strategy of integrating word segmentation and POS tagging in a single stage. This can be implemented as follows:

1) Given an input sentence, a 3tuple (special chunk, POS and word) lattice is generated by skimming the sentence from left-to-right, and looking up the word and POS lexicon to determine all the possible words and get POS tag probability distribution for each possible word.

2) Viterbi algorithm is applied to decode the 3tuple lattice to find the most possible POS tag sequence.

3) In this way, the given sentence is segmented into words with POS tags.

The rationale behind the above algorithm is the ability of HMM in parallel segmentation and classification (Rabiner 1989).

In order to overcome the coarse n-gram models raised by the limited number of orignial POS tags used in current Chinese POS tag bank (corpus), a word clustering algorithm (Bai et al 1998) is applied to classify words into classes first and then the N (e.g. N=500) most frequently occurred word class and POS pairs are added to the original POS tag set to achieve more accurate models. For example, ADJ(<许多>) represents a special POS tag ADJ which pairs with the word class <许多>. Here, <许多> is a word class label. For convenience and clarity, we use the most frequently occurred word in a word class as the label to represent the word class.

5 Partial Parsing and Full Parsing

As discussed in section 2, obviously partial parsing can have different levels and full parsing can be achieved by cascading several levels of partial parsing (e.g. 3 levels of cascaded partial parsing can achieve full parsing for the example as shown in Figure 1).

In this paper, a certain level (e.g. l-th level) of partial parsing is implemented via a chunking model, built on the HMM-based tagger as described in section 2, with $(l-1)$-th level 3tuple sequence as input. That is, for the l-th level partial parsing, the chunking model has the $(l-1)$-th level 3tuple sequence $G_1^n = g_1 g_2 \cdots g_n$ (Here, 3tuple $g_i = c_i(p_i, w_i)$) as input. In the meantime, chunk tag t_i used in the chunking model is structural and consists of following three parts:

- **Boundary Category B**: It is a set of four values 0, 1, 2, 3, where "0" means that the current 3tuple is a whole chunk, "1" means that the current 3tuple is at the beginning of a chunk, "2" means that the current 3tuple is in the middle of a chunk and "3" means that the current stuple is at the end of a chunk.

- **Chunk Category C**: It is used to denote the output chunk category of the chunking model, which includes normal chunks and the special chunk ("."). The reason to include the special chunk is that some of POS 3tuple in the input sequence may not be chunked in the current chunking stage.

- **POS Category POS**: Because of the limited number in boundary category and output chunk category, the POS category is added into the structural tag to represent more accurate models.

Therefore, t_i can be represented by $b_i\_c_i\_pos_i$, where b_i is the boundary type of t_i, c_i is the output chunk type of t_i and pos_i is the POS type of t_i. Obviously, there exist some constraints between t_{i-1} and t_i on the boundary categories and output chunk categories, as briefed in table 1, where "valid"/"invalid" means the chunk tag sequence $t_{i-1}t_i$ is valid/invalid while "validon" means $t_{i-1}t_i$ is valid on the condition $c_{i-1} = c_i$.

	0	1	2	3
0	Valid	Valid	Invalid	Invalid
1	Invalid	Invalid	Valid on	Validon
2	Invalid	Invalid	Valid	Valid
3	Valid	Valid	Invalid	Invalid

Table 1: Constraints between t_{i-1} and t_i (Column: b_{i-1} in t_{i-1}; Row: b_i in t_i)

For the example as shown in Figure 1, we can see that:

- In the 1st-level partial parsing, the input 3tuple sequence is the 0th-level 3tuple sequence .(ADJ, 发达) .(NN, 国家) .(ADV, 也) .(VB, 存在) .(ADJ, 许多) .(NN, 问题) and the output tag sequence 1_NP_ADJ 3_NP_NN 0_._ADV 0_._VB 1_NP_ADJ 3_NP_NN, from where derived is the 1st-level 3tuple sequence NP(NN, 国家) .(ADV, 也) .(VB, 存在) NP(NN, 问题).

- In the 2nd-level partial parsing, the input 3tuple sequence is the 1st-level 3tuple sequence NP(NN, 国家) .(ADV, 也) .(VB, 存在) NP(NN, 问题) and the output tag sequence 0_NP_NN 1_VP_ADV 2_VP_VB 3_VP_NN, from where derived is the 2nd-level 3tuple sequence NP(NN, 国家) VP(VB, 存在).

- In the 3rd-level partial parsing, the input 3tuple sequence is the 2nd-level 3tuple sequence NP(NN, 国家) VP(VB, 存在) and the output tag sequence 1_S_NN 3_S_VB, from where derived is the 3rd-level 3tuple sequence S(VB, 存在). In this way, a fully parsed tree is reached.

- In the cascaded chunking procedure, necessary information is stored for back-tracing. Partially/fully parsed trees can be constructed by tracing from the final 3tuple sequence back to 0th-level 3tuple sequence. Different levels of partial parsing can be achieved according to the need of the application.

6 Experimental Results

The Chinese efficient analyser is implemented in C++, providing a rapid and easy code-compile-train-test development cycle. In fact, many NLP systems suffer from a lack of software and computer-science engineering effort: running efficiency is key to performing numerous experiments, which, in turn, is key to improving performance. A system may have excellent performance on a given task, but if it takes long to compile and/or run on test data, the rate of improvement of that system will be contrained compared to that which can run very efficiently. Moreover, speed plays a critical role in many applications such as text mining.

All the experiments are implemented on a Pentium II/450MHZ PC. All the performances are measured in precisions, recalls and F-measures. Here, the precision (P) measures the number of correct units in the answer file over the total number of units in the answer file and the recall (R) measures the number of correct units in the answer file over the total number of units in the key file while F-measure is the weighted harmonic mean of precision and recall: $F = \frac{(\beta^2+1)RP}{\beta^2 R + P}$ with $\beta^2=1$.

6.1 Word Segmentation and POS Tagging

Table 2 shows the integrated word segmentation and POS tagging results on the Chinese tag bank PFR1.0 of 3.69M Chinese characters (1.12 Chinese Words) developed by Institute of Computational Linguistics at Beijing Univ. Here, 80% of the corpus is used as formal training data, another 10% as development data and remaining 10% as formal test data.

Function	P	R	F	Speed
Word Segment.	97.5	98.2	97.8	11,000 wps
POS Tagging	93.5	94.1	93.8	

Table 2: Performances of Word Segmentation and POS Tagging (wps: words per second)

The word segmentation corresponds to bracketing of the chunking model while POS tagging corresponds to bracketing and labelling. Table 2 shows that recall (P) is higher than precision (P). The main reason may be the existence of unknown words. In the Chinese efficient analyser, unknown words are segmented into individual Chinese characters. This makes the number of segmented words/POS tagged words in the system output higher than that in the correct answer.

6.2 Partial Parsing and Full Parsing

Table 3 shows the results of 1st-level partial parsing and full parsing, using the PARSEVAL evaluation methodology (Black et al 1991) on the UPENN Chinese Tree Bank of 100k words developed by Univ. of Penn. Here, 80% of the corpus is used as formal training data, another 10% as development data and remaining 10% as formal test data.

Function	P	R	F	Speed
Partial Parsing	85.1	82.5	83.8	4500 wps
Full Parsing	77.1	70.3	73.7	2100 wps

Table 3: Performances of 1st-level Partial Parsing and Full Parsing (wps: words per second)

Table 3 shows that the performances of partial parsing and full parsing are quite low, compared to those of state-of-art partial parsing and full parsing for the English language (Zhou et al 2000a; Collins 1997). The main reason behind is the small size of the training corpus used in our experiments. However, the Chinese PENN Tree Bank is the largest corpus we can find for partial parsing and full parsing. Therefore, developing a much larger Chinese tree bank (comparable to UPENN English Tree Bank) becomes an urgent task for the Chinese language processing community. Actually, the best individual system (Zhou et al 2000b) in CoNLL'2000 chunking shared task for the English language (Tjong et al 2000) used the same HMM-based tagging engine.

7 Conclusion

This paper presents an efficient analyser for the Chinese language, based on a HMM and a single engine -- HMM-based tagger. Experiments show that the analyser achieves state-of-art performance at very high speed, which can meet the requirement of speed-critical applications such as text mining.

Our future work includes:

- Syntactic analysis of the partial/full parsing results into a meaningful intermediate form.
- Research and development of Chinese named entity recognition using the same HMM-based tagger and its integration to the Chinese efficient analyser.

Acknowledgements

Thanks go to Institute of Computational Linguistics at Beijing Univ. and LDC at Univ. of Penn. for free research use of their Chinese Tag Bank and Chinese Tree Bank.

References

Abney S. 1997. Part-of-Speech Tagging and Partial Parsing. *Corpus-based Methods in Natural Language Processing*. Edited by Steve Young and Gerrit Bloothooft. Kluwer Academic Publishers, Dordrecht.

Bai ShuanHu, Li HaiZhou, Lin ZhiWei and Yuan BaoSheng. 1998. Building class-based language models with contextual statistics. *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP'1998)*. pages173-176. Seattle, Washington, USA.

Black E. and Abney S. 1991. A Procedure for Quantitatively Comparing the Syntactic Coverage of English Grammars. *Proceedings of DRAPA workshop on Speech and Natural Language*. pages306-311. Pacific Grove, CA. DRAPA.

Collins M.J. 1997. Three Generative, Lexicalised Models for Statistical Parsing. *Proceedings of the Thirtieth-Five Annual Meeting of the Association for Computational Linguistics (ACL'97)*. pages184-191.

Feldman R. 1997. Text Mining - Theory and Practice. *Proceedings of the Third International Conference on Knowledge Discovery & Data Mining (KDD'1997)*.

Rabiner L. 1989. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *IEEE 77(*2), pages257-285.

Tjong K.S. Erik and Buchholz S. 2000. Introduction to the CoNLL-2000 Shared Task: Chunking. *Proceedings of the Conference on Computational Language Learning (CoNLL'2000)*. Pages127-132. Lisbon, Portugal. 11-14 Sept.

Viterbi A.J. 1967. Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm. *IEEE Transactions on Information Theory,* IT 13(2), 260-269.

Watson B. and Tsoi A Chunk. 1992. Second order Hidden Markov Models for speech recognition. *Proceeding of the Fourth Australian International Conference on Speech Science and Technology*. pages146-151.

Zhou GuoDong and Su Jian. 2000a. Error-driven HMM-based Chunk Tagger with Context-dependent Lexicon. *Proceedings of the Joint Conference on Empirical Methods on Natural Language Processing and Very Large Corpus (EMNLP/ VLC'2000)*. Hong Kong, 7-8 Oct.

Zhou GuoDong, Su Jian and Tey TongGuan. 2000b. Hybrid Text Chunking. *Proceedings of the Conference on Computational Language Learning (CoNLL'2000)*. Pages163-166. Lisbon, Portugal, 11-14 Sept.