KNOWLEDGE ACQUISITION AND CHINESE PARSING BASED ON CORPUS ①

Yuan Chunfa, Huang Changning and Pan Shimei
Dept. of Computer Science
Tsinghua University, Beijing, China
Fax: 861-256-2768

ABSTRACT

In Natural Language Processing (NLP), one key problem is how to design a robust and effective parsing system. In this paper, we will introduce a corpus- based Chinese parsing system. Our efforts are concetrated on: (1) knowledge acquisition and representation; and (2) the parsing scheme. The knowledge of this system is principally extracted from analyzed corpus, others are a few grammatical principles, i.e. the four axioms of the Dependency Grammar (DG). In addition, we also propose the fifth axiom of DG to support the parsing of Chinese sentences.

1. Introduction

The traditional approaches of natural language parsing are based on rewriting rules. We know that when the number of rules have already increased to a certain level, the performance of parsing will be improved little by increasing the number of rules further. So using corpus-based approach, i.e. extracting linguistic knowledge with fine grain size from corpus directly to support natural language parsing is more impressive.

In this paper we will introduce the work on Knowledge acquisition and Chinese parsing based on corpus. Our work includeds:

. Take out a total of 500 sentences from geography text book of middle school to form a small Chinese corpus.

. Because Dependency Grammar (DG) directly describes the functional relations between words, and a dependency tree has not any non-terminal nodes, DG is suitable for our Corpus-Based Chinese Parser (CBCP) particularly. We marked the dependency relations of every sentence in our corpus manually.

. Input the analyzed corpus into the computer and form a matrix file for every sentence in the corpus.

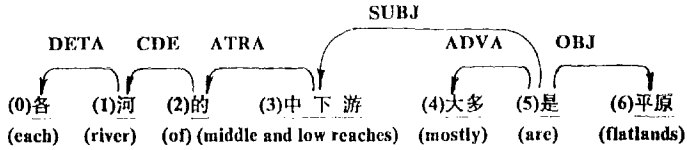
. Extract the knowledge from the matrix file and form a knowledge base.

. Implement the CBCP system for parsing input sentences and assigning dependency trees to them.

2. Construction of the knowledge base

① This project is supported by National Science Fundation of China under grant No. 69073063

At first, we marked the dependency relations of every sentences in our corpus manually. An example of analyzed sentence is as follows :



Most of the middle and low reaches of each river are flatlands.

Fig 2.1

Here: DETA(DETerminative Adjunct), CDE(Complement of "的(DE)"), ATRA(ATtRibute Adjunct), SUBJ(SUBJect), ADVA(ADVerbial Adjunct), OBJ(OBJect).

Then we run a program to input the dependency relations of every sentence to the computer and form a matrix file as bellow:

M(0 1) = DETA M(1 2) = CDE M(2 3) = ATRA M(3 5) = SUBJ
M(4 5) = ADVA M(6 5) = OBJ

In order to expound the knowledge representation, we give some definitions as below. If there are four words w1, w2, w3 and w4 with dependency relations R1, R2 and R3:

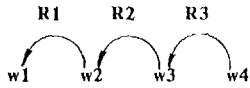


Fig 2.2

Then for the word "w3", its d–relation is R2; its g–relation is R1; and its s–relation is R3.

We extract the knowledge from the matrix file to form a frame as below :

word–name ::= [<govfreq>, <govlist>, <linklist>, <patlist>]

The slots of the frame are:

governor frequency (govfreq): It indicates that whether the given word can be a governor of a sentence and how many times it has been in our corpus.

governor list (govlist): It indicates which word can be the parent node of the given word, and what is the dependency relation between the word and its parent node. In other words, what is the word's d–relation and how many times it has occurred in the corpus. i.e.

govlist ::= [{<governor–name>{[<d–relation>, <frequency>]}*}*]

dependency link list (linklist): The d–relation and g–relation of the given words can form a pair of relations described as d–relation<——g–relation. The information on linklist includes: how many kinds of dependency links the given word have in our corpus? And what are they? how many times it has occurred? what is the position of the word's parent node (to the right or to the left of the word) in a sentence? i.e.

linklist ::= [{ <d–relation> {[<g–relation> , <position> , <frequency>]} * } *]

pattern list (patlist): The given word and its s–relations constitute a pattern of the word as: (s–relation1 s–relation2 s–relation3 ...). This pattren information describes the rationality of the syntactic structure in a dependency tree. The patlist knowledge extracted from the corpus includes: how many patterns can the word act in our corpus? What is each pattern? how many times has it occurred? What is the position (to the right or left of the word) of the children node in a sentence in our corpus? i.e.

patlist ::= [{[pattern [<frequency> , {[<s–relation> , <position>]} *]]} *]

(notes: the content inside the "{ } * " can be repeated n times, where n > 1)

3. The parser

In our CBCP system, the knowledge base will first be searched for all the possible linklist information of each word pair, according to the words in the input sentence. We use this information to construct a Specific Matrix of the Sentence (SMS). Second, remove impossible links in the SMS, and form a network. Third, we search all the possible dependency trees in the network, using the pruning algorithm. Finally, the solutions will be selected by evaluating the dependency trees. The process of removing and pruning is based on the knowledge base and the four axioms of Dependency Grammar (Robinson, J.J.1970). The four axioms are:

I. There is only one independent element (governor) in a sentence.

II. Other elements must directly depend on one certain element in the sentence.

III. There should not be any element which depends on two or more elements.

IV. If the element A directly depends on element B, and element C is located between A and B in a sentence, element C must be either directly dependent on A or B or an element which is between A and B in the sentence.

According to our Dependency Grammar practice in Chinese, we postulate the fifth axiom as follows:

V. There is no direct dependent relation between two elements which one is on the left hand side and the other is on the right hand side of a governor.

3.1 Construct a specifical matrix of a sentence

Suppose there are k words in a sentence marked as S = (w1 w2 w3 ... wi... wk), CBCP searches the linklist information of every word in the sentence. For example, if one link of wi is ATRA <——OBJ, and the link of wj is OBJ <——GOV (GOVernor) in the knowledge base, CBCP can construct the link between wi and wj as ATRA <——OBJ. The SMS will be constructed by searching all the links of words in the input sentence.

3.2 Remove impossible governors and links

Since an input sentence may form a large number of dependency trees based on the SMS, it is necessary to remove the impossible links before connecting every node to a network. Suppose in a SMS, the word A is dependent on the word B and the link between them is

Ra<——Rb. If there exists a (R1 R2 ...Ra...Rk) in B's patlist, the dependent relation of Ra<——Rb is reasonable. Otherwise, the Ra<——Rb relation is impossible, and should be removed.

The CBCP system looks for the govfreq information of each word in an input sentence. If the govfreq of a word is greater than zero, the word can be a governor. The rules of removing impossible governors are:

. If a word has no parent node in SMS, the word must be the governor (based on axiom Ⅰ). Other words which can also act as a governor must be removed.

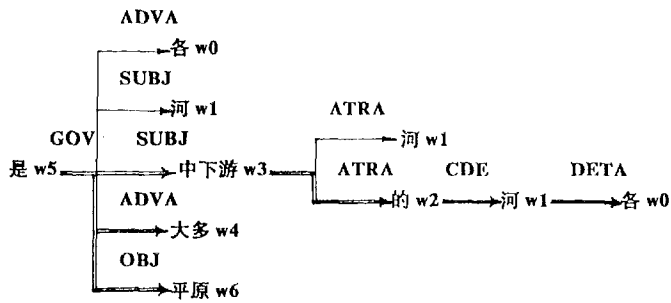
. If a word A has only one link to word B with the link Ra<——GOV, and the word B can not be a governor, the word A will not depend on any word in the dependency tree. According to axiom Ⅱ this is impossible, therefore word B must be the governor. Other words which also can act as a governor must be removed.

. When a word A has only one link to word B with the link Ra<——Rb (Rb <> GOV), and the d–relation of the word B is not Rb, the word A will not depend on any words in the dependency tree. According to axiom Ⅱ this is impossible. So the d–relation of the word B must not be the governor. Then this kind of link in which the word B is used as a governor must be removed. After removing all the impossible governors and links, the SMS of the sentence in Fig–2.1 is as follows:

M(0 1)=DETA<——CDE M(0 5)=ADVA<——GOV M(1 2)=CDE<——ATRA
M(1 3)=ATRA<——SUBJ M(1 5)=SUBJ<——GOV M(2 3)=ATRA<——SUBJ
M(3 5)=SUBJ<——GOV M(4 5)=ADVA<——GOV M(6 5)=OBJ<——GOV

3.3 Search the possible integrated tree from the specific tree

Let the governor be the root node, connecting all the nodes in order. If a node have n (n > 1) parent nodes, we can split this node to n same nodes. Let these n same nodes depend on the n parent nodes respectirely. Thus Specific Tree (ST) will be constructed. The ST of the sentence in Fig–2.1 is as bellow:



Fig–3.1

If a node appears m times in the ST, we may say the degree of freedom of this node is $\lambda = m$. If there is only one word, whose λ equals to m in a ST, then m dependency trees may be constructed. If the degree of freedom of the word–i equals to n , the degree of freedom of the word–j equals to m then the n * m dependency trees will be constructed. If there are many words with λ greater than one, the number of dependency trees being formed will be very large. Therefore, in the process of seaching an integrated dependency tree, the pruning technology must be taken. The pruning technology derives from axiom V .

After the integrated dependency trees have been produced, we use the numerical evaluation to produce the parsing result [1].

4. Experimental result and future work

When CBCP analyzed Chinese sentences in a closed corpus, it has an approximately 90% success rate (comparing with the result of manual parsing). If each word in a sentence can be found in our corpus and the corresponding dependence relation can also be found in our knowledge base, it is also feasible for CBCP to perform syntactic parsing in an open corpus.

As our research is advancing, we will enlarge the scale of our corpus and make it work on open corpus more effectively. On the other hand, we have great interests in how to retrieve more information from different aspects. For example, we want to acquire grammatical category information and semantic features for our system or equip complex feature set for each word to support corpus–based as well as rule–based system. We want to add a few rules to our system, in order to replace the frames of the words which frequently appear in our corpus. The frame of such a word is very large, but it is easy to describe its dependency relations by rules. We plan to do further research in this field.

In addition, our work can be easily expanded to set up a Chinese Collocation Dictionary. It is very difficult to make this kind of dictionary by man power, beacuase it is impossible to seek all the possible collocations of a particular word just by thinking. But it is easy to achieve this with corpus–based approach like our work. The more refined analyzing of the texts in the corpus, the more knowledge can be acquired from the corpus.

References

[1] van Zuijlen, Job M. (1990): "Notes on a Probabilistic Parsing Experiment". BSO / Language Systems, Utrecht, The Netherlands.

[2] van Zuijlen, Job M. (1989): "The Application of Simulated Annealing in Dependency Grammar Parsing". BSO / Language Systems, Utrecht, The Netherlands.

[3] 黄昌宁 (1991):《计算语言学讲义》, 清华大学.

[4] 児玉徳美 (1987):《依存文法の研究》, 研究社株式会社.