SEMANTIC AND SYNTACTIC ASPECTS OF SCORE FUNCTION

Keh-Yih SU* and Jing-Shin CHANG**

*Department of Electrical Engineering
National Tsing Hua University, Hsinchu, Taiwan, R.O.C.

**BTC R&D Center, R&D Road II, No. 28, 2nd Floor
Hsinchu Science-Based Industrial Park, Hsinchu, Taiwan, R.O.C.

Abstract

In a Machine Translation System (MTS), the number of possible analyses for a given sentence is largely due to the ambiguous characteristics of the source language. In this paper, a mechanism, called "Score Function", is proposed for measuring the "quality" of the ambiguous syntax trees such that the one that best fits interpretation by human is selected. It is featured by incorporating the objectiveness of the probability theory and the subjective expertise of linguists. The underlying uncertainty that is fundamental to linguistic knowledge is also allowed to be incorporated into this system. This feature proposes an easy resolution to select the best syntax tree and provides some strategic advantages for scored parsing. The linguists can also be relieved of the necessity to describe the language in strictly "correct" linguistic rules, which, if not impossible, is a very hard task.

Motivation

In a Machine Translation System (MTS), where the underlying grammar is large, there are many sources which may cause the system to become highly ambiguous. The system must choose a better syntax tree among all the possible ones to reduce the load of the post-editor. Some systems attack this problem by arranging the grammar rules in a descending order of their relative frequency, following the parsing paths in a depth-first manner, and selecting the first syntax tree successfully parsed as the desired one. However, rule ordering is just a locally preferred static scoring of the rule usage. Therefore, the possibility is small that the first tree selected is the correct one. Several MT systems based on the ATN formalism [Wood 70] adopt another approach. They impose condition checks to prevent the parser from trying all possible states allowed by the underlying grammar. This approach has been widely accepted and is useful in eliminating the unnecessary trials. However, there are times when legal paths are blocked inadvertently by condition checks. Therefore, the system must be tuned frequently to achieve an equilibrium between the over-generative grammar and the over-restrictive condition checks. This kind of "hard rejection" is obviously too variant and too restrictive.

A better solution is to adopt the "Truncation Strategy" (proposed by [Su 87a, 87b] for MT system) to restrict the number of parsing paths to be tried according to the relative preference of all the possible paths. The measuring mechanism of preference for the truncation strategy is called the "Score Function". It bears similarity to the select-by-preference found in other scored MT systems like the DIAGRAM grammar system [Robi 82] and METAL system [Benn 82]. Under a scoring mechanism, the parsing paths are not rejected because of the over-restrictive condition checks but rather for their low scores. This kind of "soft-rejection" prevents legal path from being blocked too early because of unsuitable condition checks. Different scoring mechanisms may be required at lexicon, syntax and semantics levels, and score can be computed during parsing or after parsing. In this paper, we propose an approach to the semantic and syntactic aspects of the score function.

Criteria for Score Function

In order to define a reasonable score function, it is essential to set up some criteria first. Eight basic criteria are listed here.

[1] The score function should reflect the absolute degree of preference of two ambiguous (sub)trees as well as their relative preferences.

[2] A good score function should be applicable either locally to a subtree or globally to a complete tree.

[3] The score function should be compositional. This means the score of a tree should be directly evaluated from the scores of its constituent subtrees.

[4] Relative rule application frequency should be included in the score function. The rule that is used most frequently should receive a higher preference.

[5] The score function should also include the semantic information embedded in the sentence, so that the semantic preference can be involved in the score function. (Since our present translation unit is a single sentence, no discourse information need to be included)

[6] The implementation of the score function should not be too complicated. In our case, it should be practical for a large-scale MT system.

[7] The database for score computation should be easy to build and easy to maintain.

[8] The preference order of ambiguous trees assigned by the score function should match those assigned by the human. In addition, the way the scores are given had better match the way that people give their preference to the ambiguous trees. (i.e. how people recognize the true meaning of a given sentence from several different interpretations)

Keeping these criteria in mind, we define a score function as follows. The score function for a subtree Xo, with derivation sequence D of $Xo(i,j) \stackrel{D}{\Rightarrow} X1(i,j_1), X2(j_1+1,j_2), \ldots Xn(j_{n-1}+1,j)$, is :

$$SCORE(Xo) = SCsyn[X1 \ldots Xn] * SCsem[(X1,KI(X1),KC(X1)) \ldots (Xn,KI(Xn),KC(Xn))]$$

In the above, Xo(i,j) is a subtree made up of terminals X1 to Xn; i to j are the word index in the sentence; and SCORE is the score of the subtree Xo. SCsyn is the unweighted syntax score. SCsem is the semantic weighting. KI is defined as the knowledge about the inherent properties of the nodes. And KC is the well-formedness condition, either syntactic or semantic, of the Xi under the given syntactic construction. To decrease the computational complexity, we can convert this multiplication equation into an addition equation with logarithmic entries.

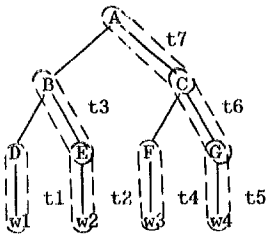
$$\log(SCORE(Xo)) = \log(SCsyn) + \log(SCsem)$$

In order to obtain the score without excessive computation and complicated algorithm, the probability model is probably one of the most common and promising approach. Under this approach, the preference measurement in a scoring mechanism can be seen as a probability assignment. The best syntax tree should be the one with highest preference probability assigned to it. This probability model can be divided into two parts. One is the syntactic score model, which is SCsyn, and the other is the semantic score model, which is SCsem. The syntactic score model uses the syntax probability as the base to generate an unweighted syntactic score for each syntax tree. The semantic score model then supplements the unweighted score with weights derived from the semantic knowledge. Incorporation of semantic information is essential for a good score function because pure syntax probability can only provide partial information for sentence preference.

Syntactic Score Model

For a syntax tree given below, we define a phrase level as a sequence of terminals and nonterminals that are being reduced at a single step of "derivation, or reduction sequence". The following example shows the reduction sequence of a bottom-up parsing. The sequence is indicated by the time series t1 ... t7 .

Example 2 :



X8 = { A }
X7 = { B, C }
X6 = { B, F, G }
X5 = { B, F, w4 }
X4 = { B, w3, w4 }
X3 = { D, E, w3, w4 }
X2 = { D, w2, w3, w4 }
X1 = { w1, w2, w3, w4 }

The unweighted score for this tree A is modeled as the following conditional probability.

SCsyn(A)
= P(X8|X7,...X1)*P(X7|X6,...X1)* ... *P(X2|X1)
≅ P(X8|X7)*P(X7|X6)* ... *P(X2|X1)
= P(A|BC)*P(BC|BFG)* ... *P(D,w2,w3,w4|w1,w2,w3,w4)
≅ P(A|BC)*P(C|BFG)* ... *P(D|w1,w2,w3,w4)

An assumption was made in the above equation. We assumed terms like P(Xi|Xi-1, Xi-2, ... X1) can be simplified into P(Xi|Xi-1). This is reasonable because at phrase level Xi-1 it will contains most of the information percolated from lower levels and needed by Xi. So, extra information needed by Xi from Xi-2 is little. We completed a simulation for testing this model and also conducted several tests on the context sensitivity of this probability model. First, we checked whether a left context (i.e. L) is relevant to the probability assignment. Using the P(X3|X2)=P(E|D,w2,w3,w4) as an example, with D as the left context of the current derivation symbol w2, we checked if P(X3|X2)=P(E|D,w2) is true? We also checked whether a right context (i.e. R) has influence on the assignment. Or is P(X3|X2)=P(E|w2,w3) true? Other test cases are LL, LR, RR, LRR, LLR, LLL, RRR, LLRR and LLLR.

Semantic Score Model

The weight-assigning process of the semantic score can be seen as an expert task where the linguist is giving the syntax tree a diagnosis. The linguist will assign a preference to a tree according to some linguistic knowledge or heuristic rules. Very often these linguistic rules are not very precise. Therefore, a good semantic score model must allow this type of inexact knowledge. Now, the problem is transformed into building a rule-based expert system that can calculate semantic scores (weightings) and handle inexact knowledge encountered during calculation. We propose a model similar to the CF model (certainty factor model) in MYCIN [Buch 85] system. It has a knowledge-rule base where each rule has a certainty factor based on the degree of belief and disbelief. The confirmation of a hypothesis then is calculated from the applicable rules and from other pieces of evidence. The CF of a hypothesis is then accumulated gradually with each additional evidence.

Each tree node will have a well-formedness factor (WFF), which is the CF for the derivation of this node, associated with it. As the knowledge, which may contain the word sense, syntactic category, attribute, etc., of leaf nodes propagates up along the syntax structure, every node's WFF will be calculated according to the rules stored in the knowledge rule-base. This WFF then becomes the semantic score of the subtree.

WFF(Xo) = SCsem[(X1,KI(X1),KC(X1) .. (Xn,KI(Xn),KC(Xn)]
where derivation sequence D : Xo $=^{D}\Rightarrow$ X1, .. Xn.

There are three major advantages of this scheme. First, linguists do not have to write a single exact rule to include all possible exceptions, because CF are given in accordance with its degree of confirmation or disconfirmation. When an exception appears, all that needs to be done is to add necessary rules and alter CF of certain existing rules. Second, the CF model simplifies the implementation of "soft-rejection" for inexact knowledge. For example, conditions (like those in ATN) can be included for disambiguation even if it is not absolute in its generality. Third, we can combine various traditional techniques in analyzing semantics with CF model to construct a uniform and flexible control strategy. This allows the inclusion of uncertain factors like semantic marker of lexicon, assignment of case role (from case grammar), and restriction of case filler. Under this control strategy, word sense disambiguation and structure disambiguation are also possible. The relative preference will be given according to the CF associated with different word sense and by the linguistic rules from the knowledge base.

All in all, with the score function defined as above, it satisfies all eight criteria we had set initially and it is a good systematic approach for assigning references to a set of ambiguous trees.

Simulation Result

A simulation, based on 1408 source sentences, was conducted to test the syntactic score model. The probability assigned to the entries ,e.g. P(E|w2,w3), in the SCsyn equation is estimated with the relative frequency of these entries. That is, we approximate P(E|w2,w3) by the ratio of the number of events {E,w2,w3} in the database and the number of events {w2,w3}. Several tests are conducted to check the influence of the context on the probability assignment. These tests include L, R, LL, LR, RR, LLL, LLR, LRR, RRR, LLRR and LLLR. Table 1 is some of the result of the simulation using sentences in the database as the test inputs.

The number of entries in the table is the number of different conditional probability, e.g. P(E|w2,w3), in the database. Each entry is assigned a probability according to its usage frequency as we explained before. The preference of a tree is the parameter that we want to estimate from these entries. If the size of database is not large enough then these probability

Table 1 : Some results of the syntactic score simulation.

size of database (sentences): 820 No. of sample test sentences = 52		
Rank	count	accumulative percentage
1	42	80.77%
2	8	96.15%
6	2	100.00%
context : LL No. of entries : 2966		
Rank	count	accumulative percentage
1	45	86.54%
2	6	98.08%
4	1	100.00%
context : LRR No. of entries : 4574		
Rank	count	accumulative percentage
1	45	86.54%
2	6	98.08%
4	1	100.00%
context : LLRR No. of entries : 6285		

size of database (sentences): 1468 No. of sample test sentences = 97		
Rank	count	accumulative percentage
1	76	78.35%
2	15	93.81%
3	3	96.91%
4	1	97.94%
6	2	100.00%
context : LL No. of entries : 4187		
Rank	count	accumulative percentage
1	83	85.57%
2	11	96.91%
3	2	98.97%
4	1	100.00%
context : LRR No. of entries : 6560		
Rank	count	accumulative percentage
1	85	87.63%
2	9	96.91%
3	2	98.97%
4	1	100.00%
context : LLRR No. of entries : 9224		

can not be approximated by the relative frequency. In general, as the size of a database increases so is the accuracy of approximation. But how big should the database be is difficult to determine. This leads us to built two databases, one having 1468 source sentences and the other having 820 sentences. If the simulation result from different base is close then we may assume that the database size is large enough. Comparing the results from these two databases, its is apparent that the size is adequate for the present simulation. Furthermore, it is also apparent that a context-sensitive scoring function must be adopted for a good preference estimation.

Two conclusions can be drawn from this simulation result. First, we should adopt three constituents in calculating the probability. The reason is that although the result of LLRR case is better than that of LRR case, the size of entries required by LLRR is considerable greater. Second, approximately 85% of syntax trees is accurately selected with only syntactic information available. Therefore, if we want to improve this result further we must include the semantic information.

Conclusion and Perspective

In a Machine Translation System, to reduce the load of the post-editor we must select the best syntax tree from a set of ambiguous trees and pass it to the post-editor. There are systems that rely on a set of ordered grammar rules or on a set of restrictive condition checks to achieve this. Unfortunately, they all have some drawbacks: one being too uncertain and the other being too restrictive. In this paper we have proposed a score mechanism for the truncation strategy to perform disambiguation during parsing. The score function, with the adoption of three context symbols, gives the power of context-sensitive grammar to an efficient context-free parser. From our simulation, the score function with just syntactic information will achieve an accuracy rate of 85%. In the near future when the semantic information is included, this accuracy rate is expected to increase. Currently, two databases, one for unweighted score computation and the other for linguistic rule base (for weighting assignment), are under the development at the BTC R&D center. After completion they will be incorporated into the truncation parsing algorithm for our third generation parser.

Acknowledgment

We would like to express our deepest appreciation to Wen-Hueh Li and Hsue-Hueh Hsu for their work on the simulations, to the whole linguistic group at BTC R&D center for their work on the database, and Mei-Hui Su for her editing. Special thanks are given to Behavior Tech. Computer Co. for their full financial support of this project.

References

[Benn 85] Bennett, W.S. and J. Slocum, "The LRC Machine Translation System," Computational Linguistics, vol.11, No. 2-3, pp.111-119, ACL, Apr.-Sep. 1985.

[Buch 85] Buchanan B.G. and E.H. Sortliffe(eds), RULE-BASED EXPERT SYSTEMS. Reading, MA: Addison-Wesley, 1984.

[Robi 82] Robinson, J.J., "DIAGRAM : A Grammar for Dialogues," CACM, vol.25, No.1, pp.27-47, ACM, Jan. 1982.

[Su 87a] Su, K.Y., J.S. Chang, and H.H. Hsu, "A Powerful Language Processing System for English-Chinese Machine Translation," 1987 Int. Conf. on Chinese and Oriental Language Computing, pp.260-264, Chicago, Ill, 1987.

[Su 87b] Su, K.Y., J.N. Wang, W.H. Li, and J.S. Chang, "A New Parsing Strategy in Natural Language Processing Based on the Truncation Algorithm",pp.580-586 Proc. of Natl. Computer Symposium (NCS) 1987, Taipei, R.O.C..

[Wood 70] Woods, W.A., "Transition Network Grammars for Natural Language Analysis," CACM, vol.13., No.10, pp.591-606, ACM, Oct. 1970.