Statistical Parsing of Messages

Mahesh V. Chitrao and Ralph Grishman

Department of Computer Science
Courant Institute of Mathematical Science
New York University

Message Processing

The recent trend in natural language processing research has been to develop systems that deal with text concerning small, well defined domains. One practical application for such systems is to process messages pertaining to some very specific task or activity [5]. The advantage of dealing with such domains is twofold - firstly, due to the narrowness of the domain, it is possible to encode most of the knowledge related to the domain and to make this knowledge accessible to the natural language processing system, which in turn can use this knowledge to disambiguate the meanings of the messages. Secondly, in such a domain, there is not a great diversity of language constructs and therefore it becomes easier to construct a grammar which will capture all the constructs which exist in this sub-language.

However, some practical aspects of such domains tend to make the problem somewhat difficult. Often, the messages tend not to be absolutely grammatically correct. As a result, the grammar designed for such a system needs to be far more *forgiving* than one designed for the task of parsing edited English. This can result in a proliferation of parses, which in turn makes the disambiguation task more difficult. This problem is further compounded by the telegraphic nature of the discourse, since telegraphic discourse is more prone to be syntactically ambiguous.

Statistical Parsing

The major objective of the research described in this paper is to use statistical data to evaluate the *likelihood* of a parse in order to help the parser prune out unlikely parses. Our conjecture – supported by our results and some prior, similar experiments – is that a more probable parse has a greater chance of being the correct one. The related work by the research team at UCREL (Unit for Computer Research on the English Language) at Lancaster University [4] and at IBM Research Lab. at Yorktown Heights [3] addressed the same problem in the context of unconstrained text. The success rate of about 50% achieved by the probabilistic parser at UCREL is certainly impressive [4], considering the fact that their corpus included a range of domains and text types. However, our objectives differ from that of UCREL in that our aim is to build a system for a specific application task (viz. information extraction). For this reason, we sacrifice breadth of coverage for higher reliability.

We use a model of probabilistic sentence generation based on a context-free grammar with independent probabilities for the expansion of any non-terminal by its alternate productions. These probabilities are adjusted to generate sentences in which the frequency of particular constructs corresponds to that observed in a sample corpus. The resulting grammar gives us not only the probabilities of sentences, but also the probabilities of alternate parses for a single sentence. It is the latter probabilities which are of primary interest to us in analyzing new sentences.

Initially, probabilities of all individual productions were determined using a corpus of text. This body of text was parsed using the NYU PROTEUS grammar which is a subset of Sager's grammar[10] based on Linguistic String Theory [7]. The parsing was performed using a chart parser [8]. The probabilities of the productions were computed by an iterative procedure described below.

The parser was then modified to use these probabilities. The probability of a parse tree is computed as the product of probabilities of all the productions used in that particular parse tree. The parsing algorithm generates parses in the best-first order [9] (i.e. most probable parse first).

The prioritizing of productions not only helps in the disambiguation of parses but also forces the parser to try more likely productions first. As a result, there is a twofold improvement attributable to this modified parsing algorithm. Firstly, the accuracy of the parsing improves since the parser provides a mechanism to identify the most likely parse. Secondly, the tendency of the parser to try more likely productions first results in a quicker conclusion of the search process and a faster parser[1].

To further enhance the performance of the system, probabilities of individual productions were computed separately for each of the contexts in which they could

[1] Speed of the parser can be approximately assessed from the number of edges generated. An edge is an intermediate hypothesis made by the parser [8]

be invoked and these probabilities were used to compute probabilities of parse trees. This modification improved the speed and accuracy of the system still further.

This paper briefly describes the methodology underlying statistical parsing and corroborates through experimental results the claim that parsing guided with statistical information is more robust and faster.

Estimation of Probabilities of Productions

In this experiment, a corpus of 105 Navy OPREP messages (prepared for Message Understanding Conference - 2 [11]) was parsed using the PROTEUS grammar. Parse trees of each sentence were accumulated separately. [2] The algorithm for estimation of probabilites of productions resembles the Inside Outside Algorithm [1]. The initial estimate of the probability of $T_{i,j}$, the j^{th} parse tree of the i^{th} sentence is

$$Pr_0(T_{i,j}) = \frac{1}{N_i}$$

where N_i is the number of parses obtained for the i^{th} sentence. We further define the current partial count of the n^{th} production of the m^{th} nonterminal attributable to $T_{i,j}$ to be

$$\hat{C}_0(P_{m,n}, T_{i,j}) = Pr_0(T_{i,j}) \times C(P_{m,n}, T_{i,j}) \qquad (1)$$

where $C(P_{m,n}, T_{i,j})$ is the actual count of the number of times this production is used in $T_{i,j}$. From this, an estimate of the probability of production $P_{m,n}$ can be found as

$$Pr_0(P_{m,n} \mid m) = \frac{\sum_T \hat{C}_0(P_{m,n}, T)}{\sum_n \sum_T \hat{C}_0(P_{m,n}, T)} \qquad (2)$$

Using these estimates, it is possible to come up with a new estimate of $Pr(T_{i,j})$ for each tree by multiplying the probabilities of all the productions used in that derivation. This enables us to reestimate $\hat{C}(P_{m,n}, T_{i,j})$, which in turn allow us to reestimate $Pr(P_{m,n} \mid m)$. This mechanism leads to an iterative process. After convergence, this iterative process gives us estimates of the probabilities of each production in the grammar.

Modified Parsing Algorithm

The parser which PROTEUS uses is a chart parser. This parser maintains a list of incomplete parses, which is called *agenda*. In the earlier parsing algorithm, partial parses were chosen in LIFO order for expansion[3]. For the sake of enabling "best first parsing", the *agenda* was converted into a priority queue (or a heap), where the priority of each incomplete parse was computed by adding the priorities of all constituent edges. This computation was performed in a recursive manner, since edges themselves are incomplete parses. Moreover, whenever an edge used a certain production, the corresponding priority (given by (3)) is also added to the priority of that edge.

The objective of the modified parsing algorithm is to generate parses in the most-likely-first order. The priority of a parse tree (finished or unfinished) is asserted to be the product of the probabilities of all the productions used in it. For a grammar with a decision tree of average branching factor 5, a parse tree that uses 20 different productions will have a priority of the order $(0.2)^{20}$ which is a small floating point number. In order to avoid dealing with such small numbers, probabilities are converted to log scale. This has the added advantage of being able to *add* probabilites instead of having to multiply them. We assign

$$Priority(P_{m,n}) = 10.0 \times log(Pr(P_{m,n})) \qquad (3)$$

This way, whenever a new edge is added to the parse tree, its priority is added to the priority of the parse tree.

Even though this modification in the parsing algorithm resulted in significant improvement in the accuracy and the speed of parsing, it was felt that the performance of the parsing could be further enhanced by extracting more specific information from the corpus. The following section describes one such scheme.

Fine Grained Statistics

In the above scheme, the amount of priority due to a production depends only on the production and not on the place where the production is used. For example, use of the production

SA → NULL

will involve a fixed amount of priority no matter where this **SA** appears. This strategy disregards the fact that **SA** appearing at certain locations may have a greater probability to use this production and **SA** appearing at some other place may have a lesser probability to use this production. Therefore, it was expected that if the probabilities for a certain non-terminal using a certain production were computed separately for each instance of each nonterminal appearing on the right hand side of a production, it will lead to a more *informed* parser. Such probabilities were obtained and the parsing algorithm was modified accordingly. This change resulted in an even better perfomance, both with respect to speed and accuracy. The following section briefly describes the further changes necessary in the parsing algorithm in order to be able to use fine-grained statistical information.

[2] The PROTEUS grammar used for this task included a number of ad-hoc scoring constraints to prefer full sentence analyses to fragments and run-ons. In accumulating parse trees, we only took the top-scoring trees for each sentence. For subsequent iterations, we used only these parse trees with new weights assigned; we did not reparse the corpus to obtain additional trees.

[3] This simply means that search space was searched in a depth-first order; there is no particular reason for this choice

Parsing Method	Correct	Misplaced Prepositional Phrase	Incorrect	Edges
No statistical parsing	34	45	61	577265
Context free statistical Parsing	47	46	47	553022
Context free statistical parsing with heuristic penalties	42	55	43	483432
Context sensitive statistical parsing	53	51	36	501357
Context sensitive statistical parsing with heuristic penalites	45	58	37	419772

Table 1: Comparison of correctness of parsing with various parsing methods

Modified Parsing Algorithm for Fine Grained Statistics

The parsing algorithm for parsing with fine-grained priorities becomes somewhat more complex. In this scheme, the priority attributable to a production is influenced by the context in which it is used. Since the priority of an edge is the sum of the priority of the production used by the edge and priorities of its constituents, the overall priority of an edge varies according to the context. This effect can be best captured by breaking up the priority of an edge into two components – the absolute component and the relative component. While the absolute component for an edge is constant, the relative component depends on the context in which the priority of this edge is being computed, since priorities of the same production are different in different contexts. These changes are discussed in greater detail in [2]

Heuristic Penalties

Even with the improvements detailed in the two prior sections, the parser suffered from a serious problem. The modified search algorithm, which is best first, ignored the length of hypotheses being compared. As a result, as the parsing progressed towards the right of the sentence, it would often *thrash* after the lengthier (and possibly *correct*) partial parses gathered more penalty than short, unpromising hypotheses near the left of the sentence. In order to have a strictly best-first search strategy, this phenomenon is unavoidable[4]. However, we studied the trade-off that existed by penalizing shorter hypotheses by a fixed amount per word [5]. This way, the *thrashing* was avoided, the parser became faster and the parser could parse several sentences which could not be parsed earlier under the same edge limit[6]. However, the parser lost the characteristic of being best first and this resulted in a slight degradation of accuracy.

Results

The statistical data was obtained from a corpus of about 300 parsed sentences. To assess the performance of various parsing algorithms, each version of the modified parser and grammar was run on a sample of 140 sentences, which were selected from the training corpus. For verification purposes, the correct regularized syntactic structures were manually garnered for each of these sentences. These were compared with the top-ranked (highest probability) parse produced by each of the various parsing schemes. The results are summarized in table 1.

Because the single most frequent source of error in a parse was a misplaced prepositional phrase, we separated out this parsing error from other errors. The table shows the number of sentences parsed correctly, the number that had one or two misplaced prepositional phrases, and the number with other errors, for each parsing strategy. It also shows the total number of edges built by the parser for all 140 sentences. Since the number of edges roughly represents the computational complexity of the parsing algorithm, it can be treated as an approximate measure of time spent.

The following conclusions can be drawn about the performance of the parser that uses the statistical information:

1. Parsing guided by statistics does better in speed as well as accuracy in comparison with the earlier algorithm.
2. The fine-grained statistics does better than ordinary statistics both in speed and accuracy.
3. The use of heuristic penalties improves the performance with respect to speed and the number of sentences parsed but accuracy suffers.

Acknowledgement

This research was supported in part by the Defense Advanced Research Projects Agency under contract N00014-K-85-0163 and grant N00014-90-J-1851 from the Office of Naval Research.

[4] The best first search expects an admissible heuristic, which by definition, always underestimates the residual penalty. We experimented with some admissible heuristics, but they were too weak to improve performance significantly.

[5] This idea is due to Salim Roukos (personal communication)

[6] Usually, a limit is imposed on the number of edges the parser can generate before abandoning the search for the parse. This occasionally results in a situation where the parser does not find an existing correct parse

References

[1] J. K. Baker, 1979, *Trainable Grammars for Speech Recognition*, Speech Communication Papers for the 97th Meeting of the Acoustic Society of America, D.H. Klatt and J.J. Wolf (eds).

[2] Mahesh Chitrao, Forthcoming, *Statistical Parsing of Messages*, Ph.D. dissertation, New York University, New York.

[3] Tetsunosuke Fujisaki, 1984, *A Stochastic Approach to Sentence Parsing*, in Proceedings of the 10th International Conference on Computational Linguistics and 22nd Annual Meeting of ACL.

[4] Roger Garside, Geoffrey Leech and Geoffry Sampson, 1987, *The Computational Analysis of English*, Longman, London, UK.

[5] Ralph Grishman and John Sterling, 1989, *Preference Semantics for Message Understanding*, Proceedings of DARPA Speech and Natural Language Workshop, Harwich Port, MA.

[6] Ralph Grishman, 1986, *Computational Linguistics, An Introduction*, Cambridge University Press, Cambridge, UK.

[7] Z. Harris, 1962, *String Analysis of Sentence Structure*, Mouton & Co., The Hague.

[8] M. Kay, 1967, *Experiments with a Powerful Parser*, Proceedings of the Second International Conference on Computational Linguistics, Grenoble.

[9] Nils Nilsson, 1981, *Principles of Artificial Intelligence*, Tioga Publishing Company, Palo Alto, CA.

[10] Naomi Sager, 1981, *Natural Language Information Processing*, Addison-Wesley, Reading, MA.

[11] Beth Sundheim, 1989, *Navy Tactical Incident Reporting in a Highly Constrained Sublanguage: Examples and Analysis.* Naval Ocean Systems Center Technical Document 1477.