Integrating Syntax and Semantics into Spoken Language Understanding[1]

Lynette Hirschman, Stephanie Seneff, David Goodine, and Michael Phillips

Spoken Language Systems Group
Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139

ABSTRACT

This paper describes several experiments combining natural language and acoustic constraints to improve overall performance of the MIT VOYAGER spoken language system. This system couples the SUMMIT speech recognition system with the TINA language understanding system to answer spoken queries about navigational assistance in the Cambridge, MA, area. The overall goal of our research is to combine acoustic, syntactic and semantic knowledge sources. Our first experiment showed improvement by combining acoustic score and parse probability normalized for number of terminals. Results were further improved by the use of an explicit rejection criterion based on normalized parse probabilities. The use of the combined parse/acoustic score, together with the rejection criterion, gave an improvement in overall score of more than 33% on both training and test data, where score is defined as percent correct minus percent incorrect. Experiments on a fully integrated system which uses the parser to predict possible next words to the recognizer are now underway.

BACKGROUND

The experiments that we report on in this paper represent some initial steps in combining speech knowledge with syntactic and semantic knowledge. These experiments have been performed using the MIT VOYAGER system [10], which provides navigational assistance for finding directions and locations of various objects (e.g., hotels, restaurants, banks) in a geographic region (Cambridge, MA). VOYAGER accepts spoken queries from untrained users and produces answers in the form of a map, written answers, and spoken output. The system has a vocabulary of some 320 words and handles questions, indirect questions, and various forms of interactive dialogue, including anaphoric reference and clarification dialogue.

In this research, we have taken an incremental approach to combining speech and language. First, we have explored how use of combined knowledge sources can influence the shape of the search space, by changing the overall scores associated with competing hypotheses. In addition, the use of combined knowledge sources can change computational efficiency by applying language constraints to predict possible next words, thus achieving significant pruning of the recognition search space. In this paper, we report primarily on experiments that change the shape of the search space; this work has been done on our loosely-coupled system using the N-best interface [11]. However, we also report briefly on the status of our experiments on the tight coupling of speech recognition and language understanding.

[1]This research was supported by DARPA under Contract N00014-89-J-1332, monitored through the Office of Naval Research.

LANGUAGE CONSTRAINTS DURING RECOGNITION

In order to obtain adequate recognition results for continuous speech recognition, it is imperative to provide some sort of language constraints. The usual approach is to adopt simple but efficient word-pair or bigram "language models," which specify the set of words that can follow a given word. Such models have the advantage of being automatically derivable from training data and computationally efficient. However, they lose any non-local language constraints and, of course, provide no linguistically relevant structural description. Furthermore, it is difficult, even when steps are taken to generalize words to their semantic category (i.e. "Boston" $\rightarrow$ all city names), to assure sufficient coverage in an independent test set. Given adequate training data, the simple word-pair or bigram language model will overgenerate, since it fails to take larger context into account. Thus the system is allowed to recognize many sentences that are ungrammatical or incoherent or inappropriate in the overall context.

The obvious solution is to bring linguistic knowledge to bear. One way is to take the best acoustic candidate and use a flexible, semantically-based phrase-spotting system to assign a meaning to the sequence of words [8]. This provides a robust interface which can ignore many recognition errors and abandons the notion of a linguistically well-formed overall sentence. It almost always produces some interpretation. However, since it adds no real linguistic constraints, it may produce many false positives (misinterpretation of the input).

A second possiblity which has been explored at some sites [1] is to have the recognizer produce a word lattice, with (acoustic) transition probabilities between words. The language system can then search this lattice for the best candidate.

Another approach, which is the baseline for these experiments, uses an N-best interface between the recognizer and the language understanding system. In this interface, the recognizer produces sentence hypotheses in decreasing order of acoustic score. The role of the language understanding system is to filter these hypotheses, choosing the first one that can be fully processed. Finally, the approach that we explore here combines a score provided by the parser (e.g, on the basis of a probability assignment), with the acoustic score, to provide a "best" answer. We will report here on the results of several experiments combining parse probabilities and acoustic score.

SYSTEM ARCHITECTURE

The VOYAGER system consists of the TINA natural language understanding system and the SUMMIT speech recognition system. These components will only be described briefly here, as they are more fully documented in [7,9,10].

TINA combines a general English syntax at the top level with a semantic grammar framework at lower levels, to provide an interleaved syntax/semantics analysis that minimizes perplexity. As a result, most sentences in TINA have only one parse. TINA uses a best-first heuristic search in parsing, storing alternate candidate parse paths while it pursues the most promising (most probable) path. In addition, the grammar is trainable from instances of parse trees, as described in the next section.

The SUMMIT system transforms a speech waveform into a segment lattice. Features are extracted for each segment and used to determine a set of acoustic scores for phone candidates. A lexicon provides word pronunciations which are expanded through phonological rules into a network of alternate pronunciations for each word. The control strategy to align the segmental acoustic-phonetic network with the lexical word-pronunciation network uses an N-best interface which produces the top N candidate word sequences in decreasing order of total path score. It makes use of an A^* search algorithm [3,4] with an initial Viterbi search serving as the mechanism for establishing a tight upper-bound estimate of the score for the unseen portion of each active hypothesis.

Language constraints include both a local word-pair constraint and a more global linguistic constraint based on parsability. The word-pair constraint is precompiled into the word network, and limits the set of words that can follow any given word, without regard to sentence context. Allowable word pairs were determined automatically by generating a large number of random sentences from the grammar [7]. The linguistic constraints are incorporated either as a filter on full-sentence hypotheses as they come off the top of the stack, or with a tighter coupling in which active partial theories dynamically prune the set of allowable next-word candidates during the search.

TRAINING PARSE PROBABILITIES

This section describes our procedure for training the probabilities in the grammar automatically from a set of parsed training sentences. From each training sentence is derived a set of context-free rules needed to parse that sentence. The entire pool of rules is used to train the grammar probabilities, with each rule occurring one or more times in the training data. By training on a set of some 3500 sentences within the VOYAGER domain, we were able to reduce the perplexity on an independent test set by a factor of three [10].

Our approach to training a grammar differs from that of many current schemes, mainly in that we have intentionally tried to set up a framework that easily produces a probability estimate for the *next word* given the preceding word sequence. We feel that a next-word probability is much more appropriate than a rule-production probability for incorporating into a tightly coupled system, since it leads to a simple definition of the total score for the next word as the weighted sum of the language model probability and the acoustic probability. While rule-production probabilities can in fact be generated from the probabilities we provide, they will not, in general, agree with the probabilities as determined by a procedure such as the inside/outside algorithm [6,2].

In our approach, the grammar is partitioned into rule sets, according to the left-hand side (LHS) category in the context free rule set. Within each partition, the categories that show up on the right-hand side (RHS) of all the rules sharing the unique LHS category for the partition are used to form a bigram language model for the categories particular to that partition. Thus the language model statistics are encoded as a set of two-dimensional tables of category-category transition probabilities, one table for each partition. A direct consequence of this bigram model within each partition is that new sibling "chains" may form, producing, in many cases, combinations that were never explicitly mentioned in the original rule set. The parser is driven only by the set of local node-node transitions for a given LHS, so that any new chains take on the same status as sibling sets (RHS) that appeared explicitly in the original grammar. While this property can at times lead to inadvertent rules that are inappropriate, it often yields productive new rules and allows for faster generalization of the grammar. Given a particular parse tree, the probability for the next word is the product of the node-node transition probabilities linking the next word to the previous word. The overall next-word probability for a given initial word sequence is then the sum over all parse trees spanning the entire word sequence.

A specific example should help to elucidate the training

process. Imagine that a training set provides a set of five rules as shown in Table 1. The training algorithm produces a transition network, as shown in Figure 1, with probabilities established by counting and normalizing pair frequencies, as would be done at the word level in a traditional bigram language model. Rule production probabilities can be regenerated from these pair transition probabilities, giving the result shown in the column "Derived Probability" in the table, to be compared with "Original Probability."

The derived probabilities are not the same as what one would get by simply counting rule frequencies. The probabilities are correct up to the point of the category NOUN; that is, there is a 2/5 probability of getting the rule group (1,4) and a 3/5 probability of getting the group (2,3,4). However, the transitions out of NOUN are conditional on the rule group. That is, rules that start with (ART NOUN) have a 50/50 chance of being followed by an ADJUNCT, whereas the remaining rules have a 1/3 chance. The method of ignoring everything except the preceding node has the effect of smoothing these two groups, giving all of them an equal chance (2/5) of finding an ADJUNCT next. This is a form of deleted-interpolation [5] and it helps to get around sparse data problems, although it is making an independence assumption: whether or not a noun is followed by an adjunct is assumed to be independent of the preceding context of the noun. In this example, no new rules were introduced. If we had, however, no training for Rule 4, it would still get a nonzero probability, because all of its sibling pairs are available from other rules. That is to say, not only does this method smooth probabilities among rules, but it also creates new rules that had not been explicitly seen in the training set.

The grammar itself includes syntactic and semantic constraints that may cause a particular next-sibling to fail. There is also a trace mechanism that restores a moved category to its deep-structure position. Both of these mechanisms disrupt the probabilities in ways that are ignored by the training method. While it is possible to renormalize on the fly by checking all next-siblings against the constraints and accumulating the total probability of those that pass, we did not in fact do this, in the interest of computation. We do plan to incorporate this normalizing step in a future experiment, to assess whether it offers a significant improvement in the probability estimates. We do currently make some corrections for the gap mechanism. Rather than using the *a priori* statistics on the likelihood of a node whose child is a trace, we simply assume that node occurred with probability 1.0. While neither of these is absolutely correct, the latter is generally much closer to the truth than the former.

The TINA grammar has been trained on more than 3500 sentences. The parse score is computed as the sum of the log probabilities of the node-node transitions in the parse tree. The probability of a given terminal is taken to be $1/K$, where K is the number of lexical items having the same lexical class.

TRAINING RULES	Original Probability	Derived Probability
1: NP = ART NOUN	1/5	6/25
2: NP = ART ADJ NOUN	2/5	9/25
2: NP = ART ADJ NOUN (repeat)		
3: NP = ART ADJ NOUN ADJUNCT	1/5	6/25
4: NP = ART NOUN ADJUNCT	1/5	4/25

Table 1: Deriving Probabilities from Training Rules.

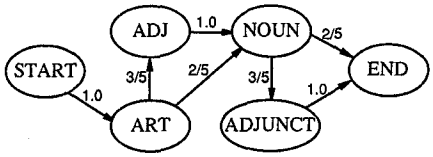

Figure 1: Probabilistic Network for NP

It would not be difficult to incorporate more sophisticated estimates of lexical items, for example, using unigram probabilities within a given lexical class, but we did not do that here, to avoid sparse data problems. The parse scores reported in the following section are the log probabilities, normalized for the number of terminals, to compensate for decreasing probabilities in longer sentences.

EXPERIMENTAL RESULTS

We have used the N-best interface with TINA as the filter for our baseline in measuring performance improvements derived from combining parse and acoustic information. To aid us in assessing the impact of various changes, we have used a composite score for system performance, computed as percent of correct answers minus the percent of wrong answers[2]. Here we define "correct answer" very strictly, namely producing the call to the VOYAGER back-end that would have been produced by a "clean" transcription of the sentence, removing false starts and filled pauses.[3] The advantage of this strict method is that the procedure can be fully automated and requires no human judgements. It does allow certain "meaning preserving" alterations, e.g, insertion or deletion

[2]This is the metric currently in use for overall performance evaluation in the DARPA Spoken Language System program.

[3]Note that this metric differs from that used to determine correctness in the results reported in the DARPA June 1990 meeting [10]. For those results, correctness was judged in terms of producing the same action, as judged by an expert. Under the new, stricter criterion, if the transcribed sentence produces no function call (action), the recognized sentence cannot possibly be correct – even if it has produced a reasonable interpretation of the input. We estimate that approximately 5% of the sentences that are incorrect here would have been judged correct under the earlier criterion. This accounts for a difference in about 10 points in score, bringing the two results into approximate agreement.

of "the" as in "the Royal East" *vs.* "Royal East". Such a criterion seems reasonable, given that correctness for a spoken language system should measure understanding, rather than word accuracy.

The N-best Interface

In the N-best interface, the grammar functions only as the filter, and N is used as a rejection criterion. As N increases, the number of correct answers increases, but the number of incorrect answers also increases. Overall system performance rises rapidly between $N = 1$ and $N = 6$, peaks at $N = 25$ and then drops off gradually, as the system finds incorrect answers at a faster rate than correct answers (see Figure 2). The optimal N is 25, with a score of 18.3 (36.1% correct and 17.8% incorrect). This figure is used as the baseline, against which performance improvements are calculated.

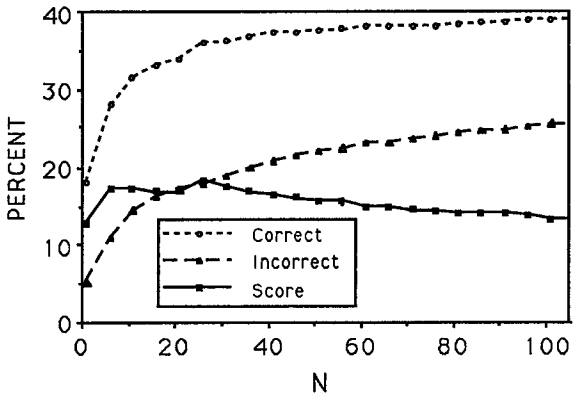

Figure 2: Performance as a Function of N. Score = Correct - Incorrect

Adding Parse Probabilities

If we now combine a parse score with the acoustic score, we get much better results. We can see how this works by looking at the example in Table 2. Here we see that the correct answer is eventually found in the N-best output (the eleventh sentence). However, it is preceded by other sentences that parse and produce possible (but wrong) function calls to VOYAGER. The N-best output produces its candidates in order of acoustic score. We see that the correct sentence has a worse acoustic score (-1336 *vs.* -1521) but its parse score is substantially better (-14.1 vs. -18.0). In general, we note that the normalized parse score is a good discriminator of right vs. wrong sentences: the mean for correct answers is -2.92 with a standard deviation of 0.75, while for incorrect answers it is -4.31 with a standard deviation of 1.78. If we compute the most obvious thing, which is a linear combination of the normalized parse score and acoustic score, it is possible, by proper choice of weight, to get the correct answer to have the best combined score. This is illustrated in Table 3, which shows the relative combined scores at two

Rank	Acoustics	Parse	#Wds	Sentence
1.	-1336	X		it i get to kendall sq
2.	-1387	-18.0	6	could i get to kendall sq
3.	-1432	-18.0	6	would i get to kendall sq
4.	-1455	X		it i'd get to kendall sq
5.	-1460	X		it i do the kendall sq
6.	-1472	X		at do i get to kendall sq
7.	-1506	X		could i'd get to kendall sq
8.	-1509	X		i'd i get to kendall sq
9.	-1511	X		could i do the kendall sq
10.	-1516	X		it i get at kendall sq
11.	-1521	-14.1	7	how do i get to kendall sq

Table 2: N-best Output With Acoustic and Parse Scores

2. could i get to kendall sq

Acoustic Score	Parse/#Wds	=	Norm Parse
-1387	-18.0/6	=	-3.0

Total Score @ W = 100 : -1387 + 100* -3.0 = -1687
Total Score @ W = 200 : -1387 + 200* -3.0 = -1987

11. how do i get to kendall sq

Acoustic Score	Parse/#Wds	=	Norm Parse
-1521	-14.1/7	=	-2.0

Total Score @ W = 100: -1521 + 100* -2.0 = -1721
Total Score @ W = 200: -1521 + 200* -2.0 = -1971

Table 3: Combining the Parse and Acoustic Scores

different weights; at weight $W = 100$, the wrong answer still has a higher combined score. However, as we increase the weight of the parse score (e.g., to $W = 200$), the correct parse receives a higher combined score. We can determine an optimal parse score weight for the training data by looking at overall score (percent correct minus percent incorrect) as a function of parse score weight. The combination that produced the optimal overall score for the VOYAGER training data was *Acoustics + 600 * Normalized-Parse*, as shown in Figure 3. In order to determine the effect of size of N on this number, we also ran experiments varying size of N. It turns out that although optimal N using only the acoustic score is $N = 25$, optimal N for the combined parse and acoustic score is 35, but it is fairly stable between $N = 25$ to $N = 100$. Using the combined acoustic plus weighted parse score, some of the original errors are corrected: the percent correct (at $N = 25$) goes up from 36.1% for the N-best case to 38.7% for the combined score, while the incorrect percent goes down from 17.8% to 15.1%. At $N = 25$, we get an overall score of 23.6%, compared to 18.3% for N-best alone (an increase of 30%).

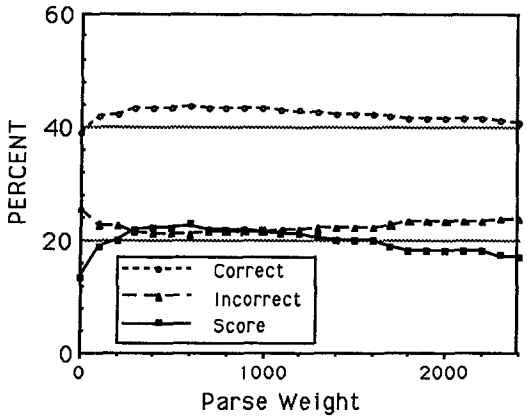

Figure 3: Performance as a Function of Weighting Factor

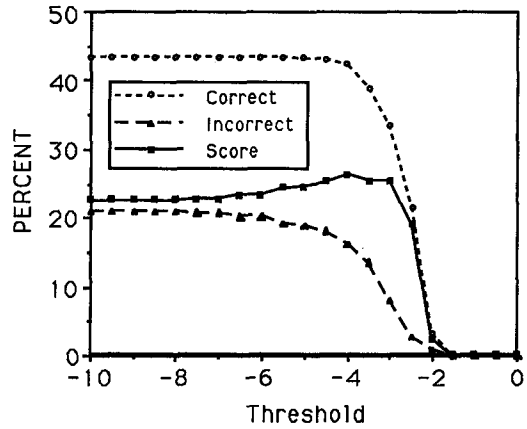

Figure 4: Performance as a Function of the Threshold

A Rejection Threshold

Finally, if we make use of the normalized parse score to formulate an explicit rejection criterion, we find that we can improve our results still further. Figure 4 shows how percent correct and percent incorrect vary with the choice of threshold. Using an empirically determined threshold of -4.0, the performance at $N = 25$ shows 37.5% correct (losing some correct answers that fall below the threshold), 10.9% incorrect (a substantial reduction from 15.1% without the use of a rejection threshold), and an overall score of 26.6% (up from 23.6% for use of combined parse and acoustic score without a rejection criterion). We also experimented with a rejection criterion based on acoustic score (e.g., difference between best score and current score) but did not find it useful in this domain; however this did turn out to be useful in the ATIS domain [12].

A comparison of the following four different configurations

```
A:  N = 1
B:  N-Best @ N = 25
C:  Weighted Parse + Acoustics @ N = 25
D:  Weighted Parse + Acoustics,
    Parse Rejection Threshold = -4.0 @ N = 25
```

	CORRECT	INCORRECT	NO ANSWER	SCORE
A	18.0	5.3	76.7	12.3
B	36.1	17.8	46.1	18.3
C	38.7	15.1	46.1	23.6
D	37.5	10.9	51.6	26.6

Table 4: Scores for Training Data

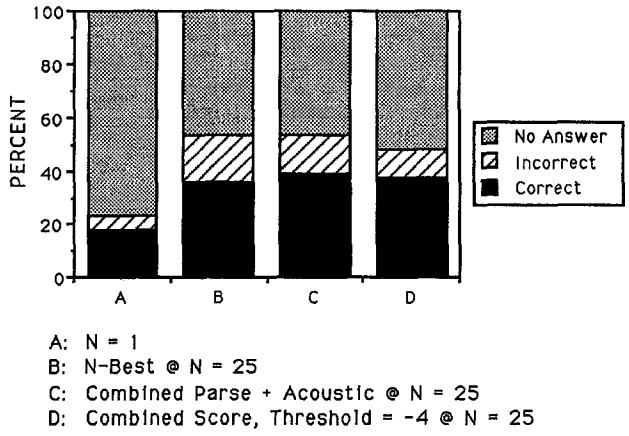

Figure 5: Overall Performance Under Four Conditions

at $N = 25$ is shown in Figure 5, with results in Table 4.

The Test Results

The overall score was optimized by running on a set of 568 training sentences (the development test set). Once we determined optimum parameters for parse score weight (W = 600), rejection threshold (T = -4), and value of N, we then ran the test data (497 sentences) using these parameters. The resulting increases in score are shown in Figure 6 for both training and test data. Overall, the test results are quite comparable to the training results. The use of a combined parse plus acoustic score resulted in an increase from 21.5 to 28.0 in overall score (30%). The use of a rejection threshold together with the combined score resulted in a small additional increase to 28.8, more than 33% over the N-best results for $N = 25$.

FUTURE DIRECTIONS

All of this research has been done as a first step towards coupling the recognizer and the language understanding system more closely. Our initial results show more than 33% improvement in score by using parse information in addition

to acoustic score. Having demonstrated that it is beneficial to change the shape of the search space using this knowledge, we are now pursuing experiments with a tightly coupled system to explore ways of increasing search efficiency. We currently have a tightly coupled version of the system running that produces the identical output but uses TINA to predict allowable next words for the recognizer, given a string of words hypothesized by the recognizer. This approach has the potential to reduce the search space for the recognizer, since it will explore only word strings that can be interpreted by TINA. This reduction in search space is done, of course, at the price of considerable computation (namely parsing the current hypotheses). We plan to investigate the trade-offs involved between the greater pruning provided by tight coupling *vs.* the greater computation required. However, our initial results are quite promising: the tightly coupled system produces its answer in under a minute, running unoptimized on a Sun SPARC-2 workstation. The next step in tight coupling will be to incorporate the parse probabilities into the overall A^* (or other) search strategy. By tuning the algorithm and off-loading some of the acoustic search to special purpose signal processing boards, we believe that the tightly coupled mode will provide improved performance over the loosely-coupled N-best interface.

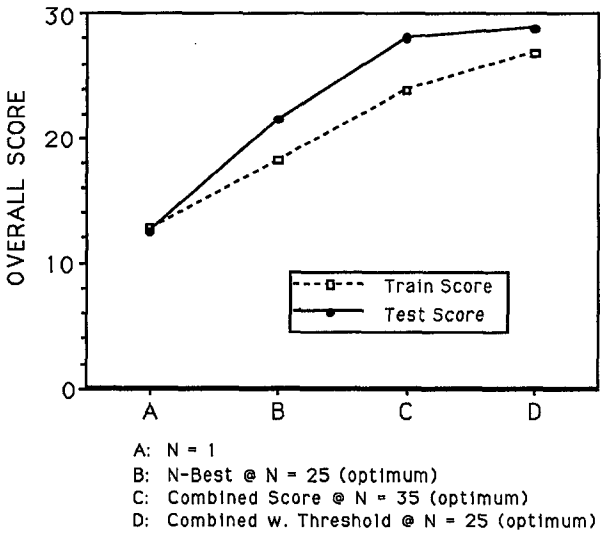

Figure 6: Performance Results Incorporating Parse Probabilities

Our results to date provide strong evidence that we can use additional knowledge from syntactic and semantic probabilities to improve overall system performance. It also indicates that explicit rejection criteria play an important part in improving system performance. In particular, the parse score threshold provides a good rejection criterion based on syntactic and semantic information. Once we develop reliable rejection criteria, we can begin to experiment with recovery strategies from rejection. For example, given a sentence that fails the rejection criterion, it might be possible to interact with the user, saying e.g., "I thought you said '...'; did I understand you correctly?" This would allow the user to confirm a correctly understood sentence and to correct a misunderstood sentence. This is surely preferable to providing misleading information on the basis of an incorrectly understood sentence. The notion of rejection criteria should also be helpful in identifying new words and sentences which contain these words. We plan to explore how to use human-machine interaction and combined syntax, semantic and acoustic knowledge to make further improvements in performance and usability of the spoken language interface.

REFERENCES

[1] Boisen, S., Y.-L. Chow, A. Hass, R. Ingria, S. Roukos, and D. Stallard, "The BBN Spoken Language System," *Proc. DARPA Speech and Natural Language Workshop*, Philadelphia, PA, February 1989.

[2] Chitrao, M., and R. Grishman, "Statistical Parsing of Messages," *Proc. DARPA Speech and Natural Language Workshop*, Hidden Valley, PA, June 1990.

[3] Hart, P., N.J. Nilsson, and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," IEEE Transactions of Systems, Science and Cybernetics, Vol. SSC-4, No. 2, pp. 100-107, 1968.

[4] Jelinek, F., "Continuous Speech Recognition by Statistical Methods," IEEE Proc., Vol. 64, No. 4, pp. 532-556, 1976.

[5] Jelinek, F., and R.L. Mercer,, "Interpolated Estimation of Markov Source Parameters from Sparse Data," In E.S. Gelsema and L.N. Kanal, *Pattern Recognition in Practice*, pp. 381-397, North-Holland, Amsterdam, 1980.

[6] Jelinek, F., "Language Modelling for Speech Recognition Using Context-free Grammars," *Proc. 3rd Symposium on Advanced Man-Machine Interaction through Spoken Language*, Tokyo, Japan, December 1989.

[7] Seneff, S, "TINA: A Probabilistic Syntactic Parser for Speech Understanding Systems," *Proc. DARPA Speech and Natural Language Workshop*, Philadelphia, PA, February 1989.

[8] Ward, W.,"The CMU Air Travel Information Service: Understanding Spontaneous Speech," *Proc. DARPA Speech and Natural Language Workshop*, Hidden Valley, PA, June 1990.

[9] Zue, V., J. Glass, M. Phillips, and S. Seneff, "The MIT SUMMIT Speech Recognition System, a Progress Report," *Proc. DARPA Speech and Natural Language Workshop*, Philadelphia, PA, February 1989.

[10] Zue, V., J. Glass, D. Goodine, H. Leung, M. Phillips, J. Polifroni, and S. Seneff, "The Voyager Speech Understanding System: Preliminary Development and Evaluation," *Proc. ICASSP-90*, Albuquerque, NM, April 1990.

[11] Zue, V., J. Glass, D. Goodine, H. Leung, M. Phillips, J. Polifroni and S. Seneff, "Integration of Speech Recognition and Natural Language Processing in the MIT VOYAGER System," *Proc.ICASSP-91*, Toronto, Ontario, May 1991.

[12] Zue, V., J. Glass, D. Goddeau, D. Goodine, L. Hirschman, H. Leung, M. Phillips, J. Polifroni and S. Seneff, "Development and Preliminary Evaluation of the MIT ATIS System," *these proceedings*, 1991.