A Spontaneous-Speech Understanding System For Database Query Applications

Roberto Pieraccini and Esther Levin

Speech Research Department
AT&T Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974, USA

ABSTRACT

In 1991 we proposed a new approach to the problem of speech understanding. The approach was called CHRONUS (Conceptual Hidden Representation Of Natural Unconstrained Speech) and was based on a stochastic representation of meaning and language. Only at the end of 1994 could we prove that a speech understanding system based on CHRONUS can be developed by a small team in a short time and still outperform traditional systems that required years of tuning. In this paper we will explain how CHRONUS allowed us to achieve two important goals: to score as the best natural language system at the 1994 ARPA ATIS evaluation and to start a new general framework for approaching the problem of language understanding.

1. INTRODUCTION

A spoken language understanding system converts speech into actions. In this specific application the input is a sequence of utterances requesting information stored in a database, and the action is the extraction of the requested database tuples. Every input sentence is interpreted in the context of the previous sentences. With the input being spontaneous speech, the problems are more related to the phenomena generally present in common spoken language rather than to the intrinsic ambiguity and complexity of language. A spontaneously spoken sentence generally contains false starts, abrupt changes of subject, ungrammaticalities, noise, but generally does not contain complex recursive forms, and ambiguities are easily resolved by the knowledge of the domain. Traditional parsers generally fail to analyze whole spontaneous sentences that cannot be interpreted according to conventional grammars. However, portions of those sentences are often grammatical and their meaning is comprehensible to a human listener. We proposed in 1991[3] a new approach that was called CHRONUS (Conceptual Hidden Representation of Natural Unconstrained Speech). CHRONUS is based on a communication paradigm that considers a spoken utterance as a noisy version of the meaning it was intended to convey. Meaning is represented by a sequence of elemental units called *concepts*. Conceptual decoding consists in detecting concepts in an utterance given the acoustic observation. With some general assumptions the model can be put in the form of a hidden Markov model (HMM [5]) whose states represent concepts and are characterized by concept conditional stochastic language models (e.g. n-grams). CHRONUS was tested several times on the ARPA ATIS [1], [2] task (a *standard* task based on an airline database) with a performance that was substantially inferior to that of more conventional systems. The reason of this was that although the conceptual decoding per se was extremely accurate [4], the inaccuracy of the other components of the system was detrimental to the overall performance. At the beginning of 1994 we started rebuilding the whole system with the following principles in mind:

- *Locality.* The analysis of a sentence as a whole is delayed as much as possible and it is demanded to a single module (the interpreter).
- *Learnability.* Everything that can be learned from available data should. A corpus is an invaluable source of knowledge that can be used for training the models used at different stages of processing.
- *Patchability.* The designer of a system should be able to introduce knowledge that either cannot be learned from data or is already available.
- *Separation among algorithms, general and task specific knowledge.* This makes the system suitable for incremental improvement.
- *Habitability.* Rather than attempting to model complex and rare linguistic events, a system has to be robust against unexpected *non linguistic* phenomena and speech recognizer mistakes.

The new system, whose functional diagram can be seen in Fig. 1, represented for us the attainment of two most important goals: the achievement of the highest accuracy in the natural language test of the 1994 ARPA ATIS [9] evaluation, and the development of a first prototype of a general toolkit for the design of speech understanding systems in the same application class. In the rest of the paper we will give a brief

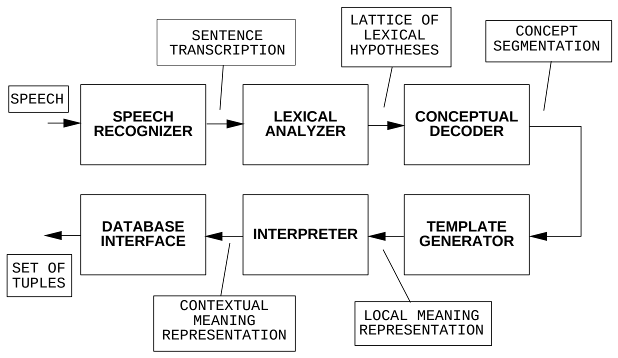

Fig. 1. A functional diagram of CHRONUS, the AT&T speech understanding system

description of the modules the current CHRONUS is made of, especially in terms of portability to other tasks, and discuss the overall performance of the system. In this paper we do not discuss the speech recognizer, whose description can be found in [7], [8].

2. THE LEXICAL ANALYZER

The function of the lexical analyzer is that of creating a lattice of word hypotheses out of a string of words (i.e. the transcription of speech into words generated by the speech recognizer [8]). The different hypotheses in the lattice correspond to possible interpretations of words and/or phrases according to predefined semantic categories. Most of the categories correspond to classes of attributes of the database. Some of them (like *numbers*) are so general that may be ported across different applications, others (like *city* or *airport*) are more specific. Function words are also merged into semantic categories. The singular and plural forms of words, for instance, constitute a category as well as different verb inflections. This both reduces the effective size of the lexicon and enhances the robustness of the stochastic conceptual representation [3], especially against recognition errors (speech recognizers are likely to confuse among different inflections of the same word). Common idioms can also be taken care of at this level, for instance phrases like *the bay area* can be substituted with *San Francisco and San Jose* by the lexical analyzer. Numbers and acronyms are grouped in all the possible ways, each one of them constituting a separate lattice hypothesis. The lexical analyzer is completely general and a new application can be developed by simply updating the vocabularies. Each vocabulary corresponds to a category (e.g. airport names, aircraft names, numbers, etc.) and each entry represents, through a regular expression, all possible idiomatic variants (e.g. *J.F.K., Kennedy Airport, Kennedy International Airport, New York City International Airport, etc.*).

3. THE CONCEPTUAL DECODER

As remarked in the introduction the idea behind the conceptual decoder is that of treating a sentence as a sequence of phrases corresponding to units of meaning called concepts. The sequence of concepts is modeled by a Markovian process (a first order process in the current implementation). The sequence of words that forms a phrase related to a given concept is also modeled by a Markovian process represented by a concept conditional n-gram language model. In the current implementation the concept conditional language models are bigram back-off networks [7]. The overall model is called conceptual HMM [3]. The function of the conceptual decoder is that of segmenting a sentence into phrases and assigning each phrase to the correspondent concept. This translates into finding the most likely sequence of states in the conceptual HMM given the lattice produced by the lexical analyzer and it is implemented as a finite state automata intersection operation as explained in [6].

There are two problems in designing a conceptual decoder for a given application: the choice of the conceptual units and the training of the conceptual HMM. The choice of the conceptual units generally requires a good knowledge of the task. Most of the units generally correspond to entities of the database. For instance, in the ATIS task there are units like `destination`, `origin`, `ground-transportation`, `aircraft_type`, `meal`, `departure_time`, `arrival_time`, etc. The level of detail of the units is generally a compromise between the resolution of the decoder and the robustness of the resulting model, given the amount of training data that is available. For instance one could choose of representing both `departure_city` and `departure_airport` as two separate concepts, being two different entities of the underlying relational database. But the structure of the phrases expressing both concepts are so similar that it does not make any sense to distinguish among them at this level. Sometimes phrases alone do not allow to distinguish among different concepts. For instance in a sentence like *show me the flights to Boston in the morning*, there are no clues on whether the phrase *in the morning* has to be related to `departure_time` or `arrival_time`, unless we use some conventional knowledge. Although the conceptual HMM could in principle learn to distinguish between the two, provided that enough training sentences of this kind are available, our choice was to define broad conceptual units (e.g. `time` and `origin`) and to demand the resolution of ambiguities to the interpretation module.

Some of the conceptual units, like `question, subject, dummy`, are quite general and related to the structure of the sentence rather than to the entities in the database.

Training a conceptual HMM implies providing a considerable amount of examples of conceptual segmentations and running a Viterbi training [5] algorithm in order to estimate the parameters of the model. Of course the most cumbersome part is providing examples of segmentations, namely segmenting by hand thousands of sentences. This can be enormously alleviated by setting up a training loop procedure [4]. The training loop consists in segmenting an initial amount of sentences (we started with about 500 sentences). The initial model estimated via the initial sentences is used in a first prototype of the whole understanding system in order to generate the answers to all the available training sentences. If an automatic evaluation procedure is available[1], it can be used for extracting from the corpus only those sentences that produced a wrong answer. Some of these sentence may have been given a wrong segmentation by the prototypical conceptual decoder that can be corrected by hand. With the assumption that the segmentation of the rest of the sentences is correct, only a small percentage of sentences have to be inspected and hand segmented. Using this procedure more than 7,000 sentences were used for estimating the parameters of the models for the ATIS task.

4. TEMPLATE GENERATOR

The template generator produces a representation of the sentence meaning in the form of a *template*, i.e. an unordered set of keyword/value pairs (tokens) [4], starting from the segmentation produced by the conceptual decoder. Most tokens correspond to attributes of the database and their values. For instance the token `DEPARTURE_CITY: SSFO` corresponds to setting the attribute `departure_city` to the value SSFO (i.e San Francisco). There are tokens that act as modifiers and are recognized either by the interpreter or by the database interface. Examples of this are the token `QUESTION` whose value `YES-NO` instructs the database interface to produce a yes/no answer, or the token `DEPARTURE_PERIOD` whose value modifies any `DEPARTURE_TIME` token in the same template to AM or PM accordingly. The information that has been requested is specified by `SUBJECT` tokens whose values have a direct correspondence to the attributes of the database.

[1]The evaluation procedure used in the ATIS project consisted in associating each answerable sentence with the set of tuples that constitute the *correct* reference answer, and comparing the output of the system with it [2]

The template generator consists of a set of programmable finite state transducers implemented as finite state machines with regular expressions and functions on their arcs. The transducers are currently programmed by hand. However they are generally extremely simple for most of the concepts. Only few concepts, like `time`, need complicate transducers, that once designed can be reused for other applications.

5. THE INTERPRETER

The function of the interpreter is to resolve the ambiguities that are present in the template. One source of ambiguity is caused by the non locality of the information: some information that modifies a concept can be present in other parts of the same sentence or in previous sentences (context dependent sentences). The modification of tokens by means of other tokens in the same sentence is performed by a set of rules most of which are of the kind:

```
if ( TEMPLATE.there_is(TOKEN1) &&
     TEMPLATE.there_is(TOKEN2) ) {
      TEMPLATE.append(TOKEN3);
      TEMPLATE.remove(TOKEN1); }
```

To deal with context dependent sentences the interpreter merges the current template with the previous one, according to merging rules defined for each class of tokens. For instance, if `DEPARTURE_TIME` is both in the current and in the previous template, only the most recent is selected to form the current query. Similarly if a new origin and destination is given in the current sentence, all the tokens corresponding to the previous sentence are deleted.

The design of the interpreter is where most of the human labor is needed. The interpretation rules are written by hand and special care should be taken in order for them not to contradict each other. A corpus and an evaluation procedure are very useful for the development of this module, since they allow to set up an automatic way of measuring the effectiveness and the consistency of each newly introduced rule. In the development of the ATIS interpreter every time new rules were introduced the system was evaluated on several corpora (about 5,000 sentences) including 1,000 sentences of the Dec.93 test set. Fig. 2 shows the percentage of correct sentences in almost 70 experimental runs during the three months in which the ATIS interpreter was developed.

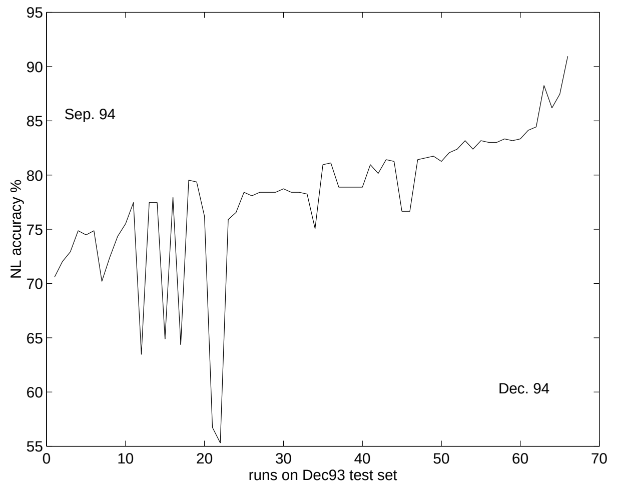

Fig. 2. Incremental performance during the ATIS system development

6. THE RELATIONAL DATABASE INTERFACE

The relational database interface takes the meaning representation in the form of a template and extracts the requested information from the database. The conventional way of doing this consists in writing a set of transcription rules that transform the template into an SQL statement. The complexity of this approach is high, and the resulting code is hard to maintain and to port to other applications. In our approach the relational database is represented by a network that can be easily derived from the database schema. The network is navigated through a *constraint propagation algorithm* that extracts the proper tuples.

7. RESULTS AND CONCLUSIONS

The system was formally tested in the official December 1994 ARPA ATIS evaluation. The performance in the three standard tests is reported in Fig. 3 and compared with that of the other sites participating in the same test.

SPREC is the accuracy of the speech recognizer [8] (% of correct words), NL is the overall percentage of correctly answered sentences of the natural language component alone (i.e. the system input was the textual transcription of each sentence), and SLS is the overall percentage of correctly answered sentences for the complete systems (i.e. the system input was the actual speech). The performance of our system in

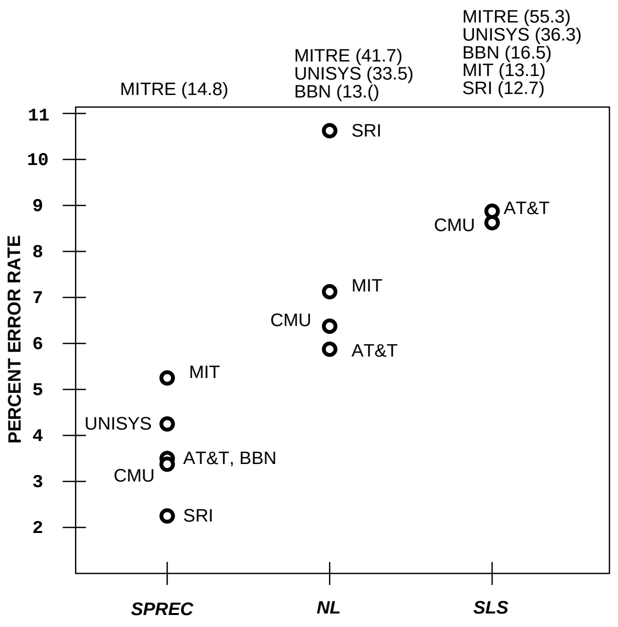

Fig. 3. Error rates reported by different sites in the 1994 ATIS test

the NL test is the highest compared to the other systems. However more important than the actual performance on the ATIS task is the flexible architecture we developed that not only allowed us to build a high performance system in a very short time, but possibly allows for a rapid development of other applications.

8. REFERENCES

[1] Price, P. J., "Evaluation of Spoken Language Systems: the ATIS Domain," *Proc. of 3rd DARPA Workshop on Speech and Natural Language*, pp. 91-95, Hidden Valley (PA), June 1990.

[2] *MADCOW*, "Multi-Site Data Collection for a Spoken Language Corpus,", *Proc. of Fifth Darpa Workshop on Speech and Natural Language*, Harriman, NY, Feb 1992.

[3] Pieraccini, R., Levin, E., "Stochastic Representation of Semantic Structure for Speech Understanding," *Speech Communication*, Vol.11 pp. 283-288, 1992.

[4] Pieraccini, R., Levin, E., "A learning approach to natural language understanding," NATO-ASI, *New Advances & Trends in Speech Recognition and Coding,* Springer-Verlag, Bubion (Granada), Spain, 1993.

[5] Rabiner L. R., Juang, B.-H., "Fundamentals of Speech Recognition," *PTR Prentice-Hall, Inc.*, 1993.

[6] Pereira, F., Riley, M. D., Sproat, R., "Weighted rational transductions and their application to human language processing," ARPA Human Language Technology Workshop, Princeton, NJ, March 1994.

[7] Riccardi, G., Bocchieri, E., Pieraccini, R., "Non Deterministic Stochastic Language Models for Speech Recognition," *ICASSP 95.*

[8] Bocchieri, E.L., Riccardi, G. Anantharaman, J., "The 1994 AT&T ATIS CHRONUS Recognizer," Proc. of 1995 ARPA Spoken Language Systems Technology Workshop, Austin Texas, Jan. 1995.

[9] Levin, E., Pieraccini, R. "CHRONUS, The Next Generation," Proc. of 1995 ARPA Spoken Language Systems Technology Workshop, Austin Texas, Jan. 1995.