Robust parsing of severely corrupted spoken utterances

Egidio P. Giachin Claudio Rullent

CSELT - Centro Studi e Laboratori Telecomunicazioni
Via Reiss Romoli 274, Torino, Italy - Ph. +39-11-21691

Abstract

This paper describes a technique for enabling a speech understanding system to deal with sentences for which some monosyllabic words are not recognized. Such words are supposed to act as mere syntactic markers within the system linguistic domain. This result is achieved by combining a modified caseframe approach to linguistic knowledge representation with a parsing strategy able to integrate expectations from the language model and predictions from words. Experimental results show that the proposed technique permits to greatly increase the quota of corrupted sentences correctly understandable without sensibly decreasing parsing efficiency.

1 Introduction

The problem addressed by this paper is how to make a speech understanding system deal with sentences for which some types of words are not recognized.

The continuous speech understanding system under development at CSELT laboratories [Fissore 88] is part of a question-answering system allowing to extract information from a data base using voice messages with high syntactic freedom. The system is composed of a recognition stage [Laface 87] followed in cascade by an understanding stage. The recognition stage analyzes speech using acoustic-phonetic knowledge. Since utterances are spoken without pauses between words, it is not possible to uniquely locate words without using syntactic and semantic constraints. Thus the actual output of the recognition stage is a set of *word hypotheses*, usually called *lattice* in the literature. A word hypothesis is characterized by its begin and end times, corresponding to the portion of the utterance in which it has been located, by a score representing its belief degree, and by the lexeme itself. The understanding stage has the task of analyzing the word lattice using linguistic knowledge and producing a representation of the meaning of the most likely consistent word sequence.

A two-stage approach to speech understanding offers several advantages and is the most widely followed in the current research. A serious difficulty, however, lies in the fact that often some short words that were actually uttered are not detected by the recognition level and hence they are missing from the lattice. To cope with this problem the understanding stage must adopt a language representation and a parsing strategy which 1) whenever possible, do not rely on such words to understand a sentence, and 2) keep parsing efficiency comparable with the case in which no word is missing. This paper describes a technique for obtaining such results. The following is divided into four sections. The next one focuses on the various implications of word undetection on the linguistic processing. Then the linguistic knowledge bases of the understanding system and the parsing strategy are outlined (assuming that all words are present in the lattice). Next the technique for coping with missing words is introduced. Finally, experimental results are discussed, showing that the proposed technique permits to greatly increase the quota of corrupted sentences correctly understandable without sensibly decreasing parsing efficiency. A discussion is also provided relating our results to other works addressing similar problems.

2 A closer examination of the problem

2.1 From the acoustical viewpoint

The phenomenology of word undetection at the recognition level is somewhat complex but mainly depends on word length. The dependency on length penalizes short words over long ones; [1] it is partly intrinsic to the signal-processing techniques used for recognition, and also heavily enhanced by coarticulation events. The consequence is that short words are frequently undetected or are given unreliable scores; then a standard parsing either would not work or would encounter heavy inefficiencies.

There is also an additional problem for continuous

[1] By 'short' word we mean a word described by one or two phonetic units. Phonetic units can be viewed approximately as phonemes [Laface 87].

This work has been partially supported by the EEC within the Esprit project 26.

speech. Often short words are erroneously detected and assigned a good score. That happens frequently when their phonetic representation is also part of a longer word that was actually uttered. For this reason the efficiency of a traditional parser would be reduced due to the necessity of taking into consideration such nonexistent words.

2.2 From the understanding viewpoint

Short words span the widest range of lexical categories and have various degrees of 'significance' (take this term informally). Some cannot be eluded and, if they are missing, it is necessary to understand the rest of the sentence and to initiate an additional interaction with the recognition level trying to figure out the most plausible words among a very limited set given by the parser; if no acceptable word is found, a dialogue with the user may be started, aimed at eliciting the essential information. Both are time-consuming operations; the latter, moreover, requires careful ergonomic considerations [Kaplan 82]. However, there are words for which the situation is not so drastic. This is the case of determiners, prepositions, and auxiliary verbs.

The treatment of words of these categories follow two main guidelines in the literature. In the former, such words act mainly as syntactic markers for multiword semantic constituents, without providing an intrinsic semantic contribution. This philosophy includes case based [Fillmore 68] and conceptual-dependency based approaches to natural language understanding [Schank 75]. In the latter guideline, such words play an independent role as semantic units and contribute compositionally with other words to the global meaning, with equal dignity [Hinrichs 86,Lesmo 85]. Clearly, given the specific problem we are addressing, it is mandatory to follow the former guideline. Happily, this commitment is coherent with the preference granted to caseframe based parsing coming from different and independent reasons inherent in speech understanding (see [Hayes 86] for an excellent discussion). The peculiar caseframe based approach summarized in the next section provides in most cases the ability of understanding a sentence without relying on such words.

3 The standard parsing strategy

3.1 Linguistic knowledge representation

Linguistic knowledge representation is based on the notion of caseframe [Fillmore 68] and is described in detail in [Poesio 87]. Caseframes offer a number of advantages in speech parsing, hence their popularity in many recent speech understanding systems [Hayes 86,Brietzmann 86], but cause two main difficulties.

First, the analysis cannot be driven by casemarkers, as is the case with written language, since often casemarkers are just those kinds of short words that are unreliably recognized or undetected at all. The standard approach is to assign to case headers the leading role, that is to instantiate caseframes using word hypotheses to fill their header slot and subsequently to try to expand the case slots. This strategy induces parsing to proceed in a top-down fashion, and works satisfactorily when headers are among the best-scored lexical hypotheses. However, it can be shown [Gemello 87] to cause severe problems if there is a bad-scored but correct header word, because the corresponding caseframe instantiation will not be resumed until all of the caseframes having better-scored but false header words have been processed. The situation of headers with bad scores happens quite frequently, especially when the uttered sentences suffer from strong local corruption due to coarticulation phenomena or environmental noise. Moreover, the standard strategy does not exploit the fact, dual to the one previously outlined, that some word hypotheses, though not being headers, have a good and reliable score. An integrated top-down/bottom-up strategy, able to exploit the predictive power of non-header words, is mandatory in such situations.

A second difficulty is given by the integration of caseframes and syntax. This is due to two conflicting requirements. From one side, syntax should be defined and developed as a declarative knowledge base independently from caseframes, since this permits to exploit syntactic formalisms at the best and insures ease of maintenance when the linguistic domain has to be expanded or changed. On the other hand, syntactic constraints should be used together with semantic ones during parsing, because this reduces the size of the inferential activity.

To overcome these problems, caseframes and syntactic rules are pre-compiled into structures called *Knowledge Sources* (KSs). Each KS owns the syntactic and semantic competence necessary to perform a well-formed interpretation of a fragment of the input. Fig. 1 shows a simple caseframe, represented via Conceptual Graphs [Sowa 84], and a simplified view of the resulting KS obtained by combining it with two rules of a Dependency Grammar [Hays 64]. The dependency rules are augmented with information about the functional role of the immediate constituents; this information id used by the offline compiler as a mapping between syntax and semantics necessary to automatically generate the KS. The KS accounts for sentences like "Da quale monte nasce il Tevere?" ("From which mount does the Tevere originate?"). The Composition part represents a way of grouping a phrase having a MOUNT type header satisfying the Activation Condition and a phrase having a RIVER type header. The Constraints part contains checks to be performed whenever the KS is operating. The Meaning part allows to generate the meaning representation starting

```
CF-24
[TO-HAVE-SOURCE]
    → (AGNT:Compulsory) → [RIVER]
    → (LOC:Compulsory) → [MOUNT]
```

```
DR-12.1
   VERB(prop) = NOUN(interr-indir-loc) <GOVERNOR> NOUN(subj)
                ;; Features and Agreements
                <GOVERNOR> (MOOD ind) (TENSE pres) (NUMBER _x) ....
                NOUN-1 ....
                NOUN-2 (NUMBER _x) ....
DR-12.2
   VERB(prop) = NOUN(interr-indir-loc) <GOVERNOR> PROP-NOUN(subj)
                .............
```

```
DefKS KS-24.12

    ;;Composition
    TO-HAVE-SOURCE = MOUNT <HEADER> RIVER

    ;;Constraints
    <HEADER>-MOUNT ((H-cat VERB) (S-cat NOUN) (H-feat MOOD ind TENSE pres ...) ...)
                   .....................
    <HEADER>-RIVER .....

    ;;Header Activation Condition
    ACTION (TO-HAVE-SOURCE)

    ;;Meaning
    (TO-HAVE-SOURCE ! * agnt 1 loc 0)
```

Figure 1: A caseframe (expressed in CG notation), two dependency rules and a corresponding KS.

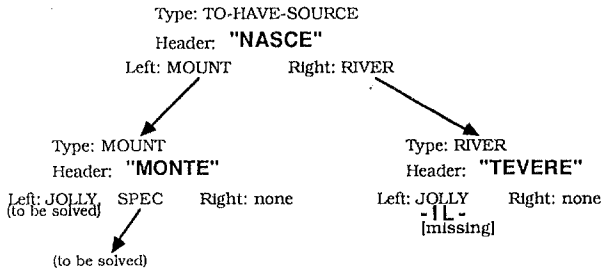

Figure 2: An example of DI.

from the meaning of the component phrases.

3.2 Parsing

Each of the phrase hypotheses generated by KSs during parsing relates to an utterance fragment and is called *Deduction Instance* (DI). DIs are an extension of the *island* concept in the HWIM system [Woods 82]. A DI is supported by word hypotheses and has a tree structure reflecting the compositional constraints of the KSs that built it. It has a score computed by combining the score of the word hypotheses supporting it. A simplified view of a DI is shown in Fig. 2. That DI refers to the sentence "Da quale monte nasce il Tevere?" ("From which mount does the Tevere originate?"); its root has been built by the KS of Fig. 1, and two more KSs were required to build the rest of it. The tree structure of the DI reflects the compositional structure of the KSs. The bottom-left part of the picture shows that there are two types (SPEC and JOLLY) that correspond to phrases that have still to be detected. Such 'empty' nodes are called *goals*. SPEC will account for the phrase "Quale" ("Which"); JOLLY represents the need of a preposition that might be missing from the lattice (this aspect is discussed later).

Parsing is accomplished by selecting the best-scored DI or word hypothesis in the lattice and letting it to be accreted by all of the KSs that can do the job. Such opportunistic score-guided search results in top-down, 'expectation-based' actions that are dynamically mixed with bottom-up, 'predictive' actions. The actions of KSs on DIs are described by operators.

Top-down actions consist in starting from a DI having a goal, and:

1. if it is a header slot, solve it with a word hypothesis (VERIFY operator);
2. if it is a case-filler slot,
   - solve it with already existing complete DIs (MERGE), or
   - decompose it according to a KS knowledge contents (SUBGOALING).

Bottom-up actions consist in creating a new DI starting either

1. from a word hypothesis, which will occupy the header slot of the new DI (ACTIVATION), or
2. from a complete DI, which will occupy a case-filler (PREDICTION).

Such a strategy is opportunistic, since the element on which the KSs will work is selected according to its score, and the actions to be performed on it are determined solely by its characteristics.

The activity of the operators is mainly concerned with the propagation of constraints to the goal nodes of each newly-created DI. Constraints are propagated from a father to a son or vice-versa according to the current parsing direction. They consist in:

- Time intervals, in the form of start and end ranges;
- Morphological information, used to check agreements inside the DI;
- Functional information, used to verify the correctness of the grammatical relations that are being established within the DI;
- Semantic type information. This information is used when, unlike the case of Fig. 1, more than one caseframe are represented by a single KS (the offline compiler may decide to do this if the caseframes are similar and the consequent estimated reduction of redundancy appears sufficiently great). In such a situation compliance with the single caseframes may have to be checked, hence the reason for this type of information.

4 Dealing with missing short words

As was pointed out, there are many different kinds of words that are short. In general, their semantic relevance depends on the linguistic representation and on the chosen domain. If the words are determiners, prepositions or auxiliary verbs, however, the integration of syntax and semantics outlined above makes them irrelevant in most cases, as very often it allows to infer them from the other words of the sentence. Such an inference may result not possible (mainly when prepositions are concerned), or the word may belong to other categories, such as connectives ("and", "or") or proper nouns, which are short but whose semantic relevance is out of question; in these cases the system must react exactly as to the lack of a 'normal' word.

Let us call 'jollies' the types of word for which only a functional role is acknowledged. Jollies are considered merely as syntactic markers for constituents to which they do not offer a meaning contribution per se. The pursued goal is twofold:

1. Parsing must be enabled to proceed without them in most cases;
2. However, whenever possible and useful, one wish to exploit their contribution in terms of time constraint and score (remember that there are also 'long' jollies, much more reliable than short ones).

The general philosophy is 'ignore a jolly unless there is substantial reasons to consider it'. The proposed solution is as follows:

1. Jollies are represented as terminal slots in the compositional part of a KS, like headers. There can be syntactic and even semantic constraints on them, but they do not enter into the rule describing the meaning representation.
2. Since we assume that jollies have no semantic predictive power, all of the operators are inhibited to operate on them.
3. Another top-down operator, JVERIFY, is added to solve jolly slots, acting only when a DI has enough support from other 'significant' word hypotheses.

Fig. 3 shows a KS deriving from the same caseframe of Fig. 1 but from a different dependency rule. Such a KS treats sentences like "Da quale monte si origina il Tevere?" ("From which mount does the Tevere originate?"), in which the word "si" is a marker for verb reflexivity.

The way JVERIFY operates depends on the result of a predicate, JOLLY-TYPE, applied on the jolly slot. JOLLY-TYPE has three possible values: SHORT-OR-UNESSENTIAL, LONG-OR-ESSENTIAL, and UNKNOWN that depend on various factors, including the lexical category assigned to the jolly slot, the temporal, morphologic and semantic constraints imposed on that slot by other word hypotheses, and the availability of such data. If the returned value is LONG-OR-ESSENTIAL, then the jolly must be found in the lattice, and its loss causes parsing to react in a way exactly similar as to the loss of any other 'normal' word. Conversely, if the value is SHORT-OR-UNESSENTIAL, the jolly is ignored by placing a suitable temporal 'hole' in the slot pf the DI. The hole has relaxed temporal boundaries so as not to impose too strict a constraint on the position of words that can fill adjacent slots; thresholds are used for this purpose. Finally, if the value is UNKNOWN, an action like the previous one is done, followed by a limited search in the lattice, looking for words exceeding the maximum width of the 'hole'. Such a search is necessary because it insures that parsing does not fail when the correct word is a jolly larger than the 'hole'. JVERIFY is submitted to the standard scheduling just as the other operators are.

5 Experimental results

The above ideas have been implemented in a parser called SYNAPSIS (from SYNtax-Aided Parser for Semantic Interpretation of Speech). SYNAPSIS is an evolution of the parser included in the SUSY system for understanding speech and described in [Poesio 87]. SYNAPSIS has been implemented in Common Lisp and relies on about 150 KSs, able to handle a 1011-word lexicon on a restricted semantic domain. An idea of the linguistic coverage is given by the equivalent branching factor, which is

```
DR-13.1
    VERB(prop) = NOUN(interr-indir-loc) REFLEX <GOVERNOR> NOUN(subj)
                 ;; Features and Agreements
                 <GOVERNOR> (MOOD ind) (TENSE pres) (NUMBER _x) ....
                 NOUN-1 ....
                 REFLEX nil
                 NOUN-2 (NUMBER _x) ....


DefKS KS-24.13

      ;;Composition
      TO-HAVE-SOURCE = MOUNT <JOLLY> <HEADER> RIVER
      ..................
      ;;Meaning
      (TO-HAVE-SOURCE ! * agnt 1 loc 0)
```

Figure 3: A KS with a jolly field.

		aux. verbs	*pron.*	*prep.*	*art.*	*refl. markers*
Parsed sentences	*skipped*	26	27	57	90	11
	present	13	1	18	0	0
Original lattices	*missing*	24	5	9	25	0
	present	15	23	66	65	11

Table 1: Jolly word detection.

missing jolly words per sentence	1	2	3
n. of sentences	40	18	4
successfully parsed	35	15	3
average n. of generated DIs	318	440	563

Table 2: Successful parsing

about 35. The system has been tested with 150 word lattices generated by processing as many sentences uttered in continuous speech with natural intonation in a normal office environment. The overall performance results in about 80% correct sentence understanding [Fissore 88].

The thresholds for JVERIFY have been experimentally determined to minimize the computational load, represented by the average number of DIs generated during each parsing. Tab. 1 shows the number of jolly words that have been skipped by the parser vs. the number of jollies actually missing in the corresponding lattices. The former figures are higher than the latter, indicating that many words, albeit present, have been discarded by JVERIFY because of their bad acoustical scores or their scarce contribution to contraint propagation.

The most apparent advantage of the above technique is the increase in the number of sentences that can be analyzed without querying the user for lacking information. Tab. 2 displays the number of lattices, corresponding to the sentences containing at least one word of jolly type, in which some of such words are missing. It is seen that about 75% of them have been successfully understood. This figure does not change substantially as the number of missing jollies per sentence increases, and hence indicates robustness. The computational load, given by the number of generated DIs, is somewhat affected by the number of missing jollies. However, this is mainly due to the fact that sentences with many jollies are also longer and syntactically complex. The actual efficiency can be better estimated from Fig. 4, where the average number of generated DIs is plot as a function of the threshold on the width of the jolly temporal 'hole'. The figure displays also the amount of parsing failures related to jolly problems (failures due to other reasons have been ignored for simplicity). The curve indicates that raising the threshold does not change much the number generated DIs (the relative oscillations of the values are small). This means that the relaxation of constraints during the application of JVERIFY is not a source of inefficiency. Moreover, there is a large range of values for which the parsing failure remains low.

The curve also shows that relaxing constraints may even speed up the parsing. This can be easily explained. When the threshold is low, no jolly is skipped, and failure occurs when jollies are missing from the lattice. When the threshold is raised, skipping begins to work: good-scored false jollies are no more a source of disturbance, and correct but bad-scored jollies are skipped thus avoiding to delay the parsing; as a consequence the overall number of DIs decreases. Further enlarging the threshold reverts this tendency, since the too-much-relaxed constraints allow the aggregation of words that would have been discarded with stricter constraints; failures occur when one of such aggregations makes up a complete parse scoring

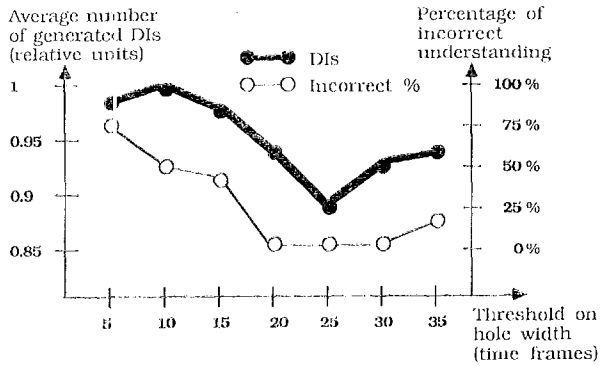

Figure 4: Performance vs. width threshold.

better than the correct one.

6 Conclusions and links with current research

Experiments show that the presence of jolly slots solvable as described above, beside permitting to successfully analyze a much greater quota of word lattices, also speeds up parsing preventing it from being misled by false jollies. This well compensates for the growth of the inferential activity due to the relaxed temporal constraints in the DIs containing 'holes'. As a consequence it is possible to use KS having chains of two or even three adjacent jolly slots without compromising excessively the global performances. This is a novel improvement over systems that, to our knowledge, only admit one single skippable word and use a more rigid linguistic knowledge representation [Tomita 87] or recognize any configuration of missing words but do not distinguish cases in which the information content of an absent word can be ignored [Goerz 83].

An attracting feature of the present parsing technique is that the KS activities are modularized into a set of operators. Consequently, it remains open to 'local' improvements on single operators as well as to overall heuristic adjustments on the score-guided control strategy. As an example, the response of the predicate JOLLY-TYPE of the operator JVERIFY may be rendered more 'intelligent' by exploiting further information, such as estimates of the expected word length, that has not been kept into consideration in the present implementation.

A different philosophy arising in very recent speech understanding research developments entrusts the problem of solving troublesome portions of the utterance (including those were jollies were not found) to a deeper acoustical analysis guided by linguistic expectation [Niedermair 87]. Our approach is not in conflict, but rather complementary to it. We believe that combining the two approaches would lead to a research area that should turn very fruitful in producing robust speech parsing.

The authors wish to express their gratitude to their colleague, the late Dr. Susa, for his contribution to the development of the system.

References

[Brietzmann 86] A.Brietzmann, U.Ehrlich, "The role of semantic processing in an automatic speech understanding system", *Proc COLING 86*, Bonn.

[Fillmore 68] C.J.Fillmore, "The case for case", in Bach, Harris (eds.), Universals in Linguistic Theory, Holt, Rinehart, and Winston, New York, 1968.

[Fissore 88] L. Fissore, E. Giachin, P. Laface, G. Micca, R. Pieraccini, C. Rullent, "Experimental results on large-vocabulary speech recognition and understanding", *Proc. ICASSP 88*, New York.

[Gemello 87] R. Gemello, E. Giachin, C. Rullent, "A knowledge-based framework for effective probabilistic control strategies in signal understanding", *Proc. GWAI 87*, Springer Verlag ed.

[Goerz 83] G.Goerz, C.Beckstein, "How to parse gaps in spoken utterances", *Proc. 1st Conf. Europ. Chapt. ACL.*

[Hayes 86] P.J. Hayes, A.G. Hauptmann, J.G. Carbonell, M. Tomita, "Parsing spoken language: a semantic caseframe approach", *Proc. COLING 86*, Bonn.

[Hays 64] D.G.Hays, "Dependency theory: a formalism and some observations", Memorandum RM4087 P.R., The Rand Corporation.

[Hinrichs 86] E.W.Hinrichs, "A compositional semantics for directional modifiers", *Proc. COLING 86*, Bonn.

[Laface 87] P.Laface, G.Micca, R.Pieraccini, "Experimental results on a large lexicon access task", *Proc. ICASSP 87*, Dallas.

[Lesmo 85] L.Lesmo, P.Torasso, "Weighted interaction of syntax and semantics in natural language analysis", *Proc. IJCAI 85*, Los Angeles.

[Kaplan 82] S.J.Kaplan, "Cooperative responses from a portable natural language query system", *Artificial Intelligence 19*, 1982.

[Niedermair 87] G.T.Niedermair, "Merging acoustics and linguistics in speech understanding", *NATO ASI-Conference*, Bad Windsheim.

[Poesio 87] M.Poesio, C.Rullent, "Modified caseframe parsing for speech understanding systems", *Proc. IJCAI 87*, Milano.

[Schank 75] R.Schank, Conceptual Information Processing, North-Holland, New York, 1975.

[Sowa 84] J.F.Sowa, Conceptual Structures, Addison Wesley, Reading (MA), 1984.

[Tomita 87] M.Tomita, "An efficient augmented-context-free parsing algorithm", *Computational Linguistics*, Vol. 13, n.1-2, Jan-June 1987.

[Woods 82] W.A.Woods, "Optimal search strategies for speech understanding control", *Artificial Intelligence 18*, 1982.