A MIXED APPROACH TO SPEECH UNDERSTANDING

*Mauro Cettolo *, Anna Corazza *, Renato De Mori ***

* IRST–Istituto per la Ricerca Scientica e Tecnologica
I–38050 Povo, Trento, Italy
** School of Computer Science McGill University
3480 University str, Montreal, Quebec, Canada, H3A2A7

ABSTRACT

This paper presents a mixed approach to spoken language understanding that tries to make best use of the advantages of both statistical and knowledge-based algorithms. Results obtained on ATIS (Air Travel Information System) scenario transferred to Italian language are presented and discussed.

1. INTRODUCTION

In Spoken Language Understanding (SLU), in addition to difficulties intrinsic to natural language processing, both recognizer and user errors must be handled. Nevertheless, for a lot of applications using a human-machine speech based interface, not all semantic contents of the utterance are relevant to the communication. The ATIS (Air Travel Information System) task is an example of such an application. In this scenario, the user asks information about North American flights and ground transportation connecting airports to downtown. For the purpose of the task, only the type of user request and a few constraints are relevant, while the rest of the utterance contents can be ignored.

Given the goal of extracting from the input utterance just the information needed to complete the query, two opposite kinds of approaches are possible: *statistical* and *knowledge-based*. Both of them have advantages and disadvantages. Statistical approaches [7] need a lot of labelled data (only supervised learning is considered here), but require very little a-priori information and extract all the information necessary to fulfil the task from data. They are easily adapted to new domains and tend to be robust with respect to spontaneous speech phenomena and to recognizer errors. On the other hand, knowledge-based approaches [2] do not need data collection and labelling, but require a lot of experts' work to put the necessary knowledge into the system. Since the knowledge is hand-coded, care must be taken to keep it consistent and error-free.

This paper presents a mixed approach to SLU that tries to make best use of the advantages of both approaches. The application domain is the ATIS task transferred to the Italian language, keeping the original database.

In SLU, error sources are various and very difficult to model. In fact, spontaneous speech phenomena are very hard to predict and lower the recognizer's performance in an unpredictable way. Therefore, the best way to test the presented algorithm is to process data collected by simulation through a baseline of the complete system. This paper briefly sketches the architecture of the system implemented at IRST, focusing on the understanding module, and on its experimental evaluation. A discussion of the results concludes the paper.

2. SYSTEM ARCHITECTURE

The system architecture is depicted in Figure 1. The modules are in a pipeline and each of them performs a well-defined function in the process of transforming the speech signal into the SQL query. The modules communicate with each other by standard UNIX pipe mechanism. It is an only software system, except for the AD/DA converter for which the standard audio server of workstation or PC is used. Every module presented in the following is implemented in *C* or *C++*, unless otherwise specified.

The *recognizer* maps the input speech signal into the corresponding sequence of words.

The *preprocessor* recognizes certain basic semantically relevant types such as dates, times, names of cities, airports, airlines. It is implemented by using the pair of UNIX tools *lex-yacc*, which allows compiling parsers for LR grammars.

The goal of the *understanding module* is to extract the meaning from the text processed by the preceding modules, that is to understand what the user wants to know. It uses binary classification trees and a simple expert system built by means of CLIPS, the tool released by NASA for expert system development.

The *SQL query generator* builds the SQL query starting from the semantic frame representation of the sentence, in order to allow access to the ATIS database. For its implementation, the TXL[1] public domain software was adopted.

3. THE RECOGNIZER

The recognizer is derived from that developed at IRST for large vocabulary dictation tasks (20K words). 48 context and speaker independent phonetic units are modelled by left-to-

[1] TXL 7.4, (c)1988-1993 Queen's University at Kingston.

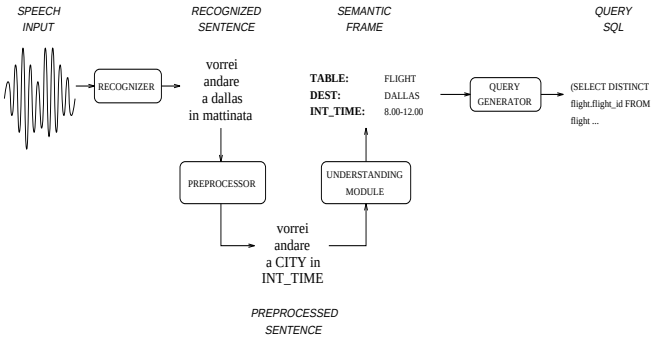

Figure 1: System architecture.

right HMMs with three or four states [1]. The output is the one-best sentence.

On the 1250-word vocabulary, a Shift-β bigram Language Model (LM) was estimated [5]. Intra-words acoustic constraints (phonetic transcriptions) and inter-words linguistic constraints (language model) are compiled into a sharing-tail tree-based network which defines the search space at unit level for the decoding algorithm [4].

Since very little data is available for LM training, ten word classes were defined: *airline-name*, *airport-name*, *city-name*, *day-name*, *meal-description*, *month-name*, *state-name*, *-1unita'-* (numbers between "ten" and "nineteen"), *-decine-* ("twenty","thirty",...,"ninety"), *-unita'-* ("one",...,"nine"). For about sixty English words (mainly city and airline names), multiple transcriptions were supplied.

4. THE PREPROCESSOR

Let a set of *classes* be defined, each related to a simple semantic concept such as times, names of cities, airports, airlines. The goal of the preprocessor is to recognize substrings corresponding to class instances inside the input sentence and replace them with the class label. The value is stored for successive use. The analysis is left-to-right.

In the following, some significant examples of time interval that the module has to be able to find are reported. They are translated from Italian into English to facilitate reading.

Preprocessor input	Recognized instances
...leave at half past twenty two	[22.30-22.30]
...leave after half past twenty two	[22.30- –]
...leave before half past twenty two	[– -22.30]
...leave between half past twenty two and a quarter to midnight	[22.30-23.45]

In these examples there are a lot of problems to face. First, the sequences of words corresponding to single numbers have to be recognized ("twenty two"); then, the value of the number has to be determined ("22"); the number has to be linked to close numbers or words in order to discover that together they define an hour ("22.30"); finally, the relationship between the hour and the near words has to be discovered to find out interval bounds explicitly mentioned by the user. Completing the instances with missing information (e.g. implicit or default interval bounds) is left to the understanding module.

5. THE UNDERSTANDING MODULE

The goal of the understanding module is to convert the preprocessed string of words into a semantic frame. It needs some knowledge about the domain, which is obtained in two ways: from the data through a statistical algorithm, *Binary Classification Trees* (BCTs) [3, 6], and from the expertise of the designer, through a *Rule Based Module* (RBM). The statistical approach is very expensive because a lot of labelled data are needed. However, it is more robust in dealing with spontaneous speech input and with the recognizer's errors. On the other hand, a pure rule-based approach requires that an expert define all the rules necessary to a translation. This may be less expensive than collecting and labelling all the data necessary to a completely statistical approach, but it is in any case time-consuming, strictly domain dependent and more error prone, because decisions are very often not consistent if they have been made by two different persons or even by the same person in different periods. For these reasons, it seems convenient to use both approaches, trying to find which aspects of the problem can be modelled by statistics, and which by rules. In fact, the amount of knowledge to be put into the RBM is to be decided on the basis of the amount of data available and the ability of the BCTs to extract the rest of the information: the RBM at least must be able to deal with all the phenomena for which the data are insufficient to properly train the statistical part.

The semantic representation is frame-based: it includes a *frame type*, which represents the query main object (e.g. flight, fare, ground transportation, ...), and some *slots*, representing the constraints the query goal has to satisfy (e.g. origin, destination, kind of meal served, class fare, ...). For every frame type a collection of constraints is possible: in ideal conditions some of them are mandatory while some others are optional. In this work, all slots are optional because acoustic recognition errors can make it impossible to extract a constraint. Nevertheless, if only a constraint is missing, the data given by the database will include the requested information, even if some useless data will be added to it. This strategy in handling error seems to be more acceptable to the user than simply refusing the query.

The translation from the semantic representation into an SQL query to be sent to the data base is a one-to-one translation. The only difficulty is that it requires a syntactical analysis of the input and the application of transformation rules on the syntactic tree.

In Figure 2, the organization of the understanding module is depicted: the RBM can be considered as a small expert system that also uses a statistical knowledge source (BCTs). The first step of the RBM is to assert all the facts that can be

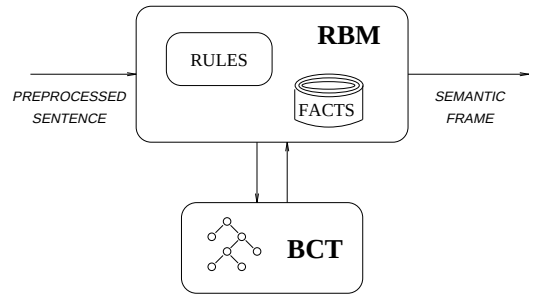

Figure 2: Understanding module architecture.

derived from the sentence. In some cases they can be directly derived from the input, in some others by asking the BCT module for a classification. The facts collected in this way are then processed by an inferential engine based on rules, whose goal is to check the consistency between the frame type and the asserted slots. After having solved conflicts using a default strategy, a frame is output, which includes a frame type and a list of consistent slots.

5.1. Binary Classification Trees

The algorithm that associates a frame type or a slot to each input sentence is a classifier, in which the possible classes are represented by all the frame types or the possible slots. BCTs are the classification tool used for this task. They are a particular case of CARTs (Classification and Regression Trees), extensively considered in [3].

A BCT is a binary tree in which each internal node has a label with a YES/NO question and two children, one for a positive answer to this question and one for the negative one. With these conventions, for each input sentence a path starting from the root and ending in a leaf is defined: at each step the path moves from the current node to one of its children on the basis of the answer to the question applied to the input sentence. A class is associated to each leaf: the output is given by the class associated to the end of the path.

The set of possible questions is represented by the set of all possible *keywords*. Every word (having a sufficient number of occurrences) can be a keyword; the question is whether or not a given keyword appears in the input sentence. Obviously, keywords are very simple questions and are not able to take into account sophisticated syntactic constructions. Nevertheless, they can be applied very efficiently and the preprocessing step can substitute every interesting syntactic construction by a keyword, which can then be used by the BCT.

6. EXPERIMENTS

6.1. Corpus

The corpus consists of 3485 ATIS-3 sentences translated into Italian and of 180 spoken sentences uttered by 20 speakers (10 female) acquired by simulations. The same speakers were asked to read a total of 195 sentences chosen among those translated of class A[2]. Recorded sentences, 195 read and 180 spoken, were used for testing purposes.

For LM training, the 3290 translated sentences (without those read) and the 375 test sentences (see Subsection 6.2) were used. Words within each class defined in Section 3 were assigned the same probability.

For BCT training, among the 3290 translated sentences (still without those read), only 1432 class A sentences referring to one of the five frame types *airline*, *airport*, *fare*, *flight* and *ground_service* were chosen. Another 142 sentences were written by hand. The distributions in terms of frame types for both the corpus used for BCT training and the 375 test sentences are reported in Table 1.

Frame Type	training		test	
	#	%	#	%
airline	143	9.1	26	6.9
airport	37	2.3	3	0.8
fare	146	9.3	39	10.4
flight	1198	76.1	291	77.6
ground_service	50	3.2	16	4.3
total	1574	100.0	375	100.0

Table 1: Corpus for BCT training and test: distribution in terms of frame types.

6.2. Recognition Experiments

Recognition experiments were performed by adopting a Leaving-One-Out (LOO) technique on the speakers for LM training. That is, for each speaker an LM was trained on the fixed 3290 translated sentences and on the test sentences of the other nineteen speakers, and the test was done on the read and spoken sentences of the current speaker. Recognition rates in terms of Word and Sentence Error Rate (WER and SER) are reported in Table 2.

	WER	SER
read	9.24	53.85
spoken	17.94	82.22
total	14.06	67.47

Table 2: Word and sentence error rate of the recognizer.

6.3. Classification Experiments

This experiment regards the classification by a BCT of preprocessed (Section 4) sentences in terms of frame types. Also in this case, a LOO technique on speakers was adopted. At each iteration, a BCT was trained on the fixed 1574 sentences (see Table 1) and on the test sentences of the other nineteen speakers (both transcription and recognized text), and the test sentences of the current speaker were classified. Classification results both of transcriptions and recognized texts are reported in Table 3.

[2] In ATIS class A sentences are context independent.

	transcription		recognized	
	#error	%error	#error	%error
read	10	5.1%	17	8.7%
spoken	13	7.2%	21	11.7%
total	23	6.1%	38	10.1%

Table 3: Classification of transcription and recognized texts.

The precision of the classification for each frame type is shown in Table 4.

	transcription		recognized	
airline	24/26	92.3	20/26	76.9
airport	0/3	0.0	0/3	0.0
fare	35/39	89.7	31/39	79.5
flight	279/291	95.9	274/291	94.2
ground_service	14/16	87.5	12/16	75.0
total	352/375	93.9	337/375	89.9

Table 4: Precision of the classification.

6.4. Understanding Experiments

Here, the whole system is tested. The 375 test sentences were semantically labelled (frame = frame type + constraints) by hand. An error occurs when there is at least one difference between the hypothesized frame and the reference frame. Semantic recognition rates are reported in Table 5, both for Natural Language System (NLS), i.e. on transcription of sentences, and for Spoken Language System (SLS), i.e. on recognized texts.

	transcription		recognized	
	#error	%error	#error	%error
read	13	6.7%	45	23.1%
spoken	18	10.0%	59	32.8%
total	31	8.3%	104	27.7%

Table 5: NLS and SLS errors.

Since consequences of semantic errors are different depending on their type, the given definition of semantic error is quite severe. A possible relaxation is as follows: "dangerous" semantic errors are those for which the set of data satisfying the true request is not included in the set of data satisfying the hypothesized request. By using this definition, figures of Table 5 become those in Table 6.

7. DISCUSSION

The results presented in the preceding section suggest some observations. First of all, the recognizer performance is very different on the read sentences and on the spoken sentences. The main reasons are two: firstly, acoustic units were trained on read material and no strategy toward spontaneous speech problems was applied; secondly, the data were probably not enough to properly train the language model.

	transcription		recognized	
	#error	%error	#error	%error
read	11	5.6%	29	14.9%
spoken	15	8.3%	40	22.2%
total	26	6.9%	69	18.4%

Table 6: NLS and SLS dangerous errors.

Due to the data scarcity, only the frame type could be identified using the BCTs. The rest of the semantic contents, i.e. the slots, are extracted by the RBM.

The importance of the recognizer's performance is shown by the figures in Table 5. Nevertheless, Table 3 proved BCTs' robustness with respect to the recognizer's errors, even if in the authors' opinion, the training material was still too little to completely exploit the algorithm's potential. On the other hand, the remaining SLS semantic errors were more due to the recognizer than to the hand-written rules used to extract slots. This means that no definitive conclusion may be derived on the robustness of the knowledge based approach.

Finally, it should be observed that system performance is not easy to evaluate. In fact, a different weight should be given to the different errors on the basis of their influence on the overall behavior of the system.

8. REFERENCES

1. B. Angelini, F. Brugnara, D. Falavigna, D. Giuliani, R. Gretter, and M. Omologo. Speaker independent continuous speech recognition using an acoustic-phonetic italian corpus. *Proc. of ICSLP*, pages III:1391–1394, 1994.
2. S.K. Bennacef, H. Bonneau-Maynard, J.L. Gauvain, L. Lamel, and W. Minker. A spoken language system for information retrieval. *Proc. of ICSLP*, pages III:1271–1274, 1994.
3. L. Breiman, J.H. Friedman, R.O. Olshen, and C.J. Stone. *Classification and Regression Trees*. Wadsworth, Pacific Grove, Cal., 1984.
4. F. Brugnara and M. Cettolo. Improvements in tree-based language model representation. *Proc. of EUROSPEECH*, pages III:1797–1800, 1995.
5. M. Federico, M. Cettolo, F. Brugnara, and G. Antoniol. Language modeling for efficient beam-search. *Computer Speech and Language*, 9:353–379, 1995.
6. R. Kuhn, R. De Mori, and E. Millien. Learning consistent semantics from training data. *Proc. of ICASSP*, pages II:37–40, 1994.
7. R. Pieraccini and E. Levin. *A learning approach to natural language understanding*, volume NATO ASI Series, F147. Springer-Verlag, Berlin Heidelberg, 1995. Antonio J. Rubio Ayuso and Juan M. Lopez Soler editors.