AN INTER-DOMAIN PORTABLE APPROACH TO INTERCHANGE FORMAT CONSTRUCTION

A. Corazza
ITC-irst
Via Sommarive, I-38050 Povo/Trento, Italy
corazza@itc.it

6th European Conference on
Speech Communication and Technology
(EUROSPEECH'99)
Budapest, Hungary, September 5-9, 1999

10.21437/Eurospeech.1999-531

ABSTRACT

The work presented in this paper regards the construction of the formal representation of the content of the input sentence. It is part of a speech-to-speech translation system, where this language independent format is used as an intermediate representation between source and target languages. The general application domain is travel planning, but it is divided in more specific domains, such as hotel and transport reservation. Starting from the word string output by the recognizer, the construction of the content representation requires the recognition of: 1) the speech act and concept list, and 2) a list of arguments in the form of argument name/value pairs. For the former, semantic classification trees [5] are used and domain portability depends on the availability of labeled data for the new domain. For argument extraction a pipeline of filters is used, where each filter is based on a semantic grammar, which in most cases is regular. Then, these filters can be used all together for the most general analysis, or separately when only one or a few domains are considered. Argument extraction is described and evaluated in terms of the changes in precision and recall with respect to the number of filters and of rules.

1. INTRODUCTION

A crucial point for designing effective speech based systems is the possibility of porting them to new domains or new versions of the same domain. For systems making use of domain dependent information, this requires algorithms and methodologies which allow a rapid adaptation of all the modules to the new domain.

The work here presented was developed and tested in the context of the C-Star Consortium [8]. The application domain considered is particularly suited for studying domain portability, since the general domain (travel planning) has been divided in different specific domains, such as HOtel REservation (HORE) and TRansport INformation (TRIN) and others which will be considered in the future. The application scenario considers a multi-lingual dialogue between a travel agent and a customer, each talking in his/her own mother tongue. It is a typical negotiation application, where one of the two partners offers a service or a product to a potential buyer.

Some of the partners in the consortium are collaborating on a common architecture based on an intermediate representation, called Interchange Format (IF) [6]. The IF gives a formal representation of the content of the utterance which has to be translated. For every considered language an analysis module builds such intermediate representation starting from the sentence uttered by a speaker and a synthesis module synthesizes the text generated from the intermediate representation.

In the following section the IF is briefly described; a more complete description and discussion can be found in [6]. In Section 3 the problem of its construction is addressed, with particular attention to argument extraction. Evaluation is presented in Section 4 and discussed in Section 5.

2. IF DESCRIPTION

The IF assigns a label to each semantic unit of the input utterance. This label is composed by four levels: 1) a label designing the speaker role in the conversation ("a:" stands for agent, "c:" for customer); 2) the speech act (e.g., greeting, request-information, accept); 3) a list of concepts, describing the focus of the segment (e.g., hotel, train, availability, price); 4) a list of arguments, composed by an argument name and a value (e.g., time=monday, room-type=single).

Not all the levels are equally affected by a domain change. The speaker and the speech act do not depend on the domain but only on the application. The domain influence begins to be relevant at the concept and the argument levels, which are strictly correlated. Some of the concepts are very general: *temporal*, *numeral*, *location* are expected to be present in all the domains. Others depend more on the fact that the scenario concerns a commercial transaction: *price*, *payment*, *availability*. Finally, some of them are strictly dependent on the domain: *hotel*, *room* in the HORE domain; *flight*, *train*, ... in the TRIN domain. Similar considerations can be made for the arguments.

3. THE IF CONSTRUCTION

The speaker's role is easily determined. The other three IF levels have to be determined on the basis of the output of the acoustic module. It should be noted that semantic segmentation is performed before the IF construction [7], on the basis of only lexical and acoustic information. Therefore, the acoustic module produces semantic segments, composed of words and extra-linguistic phenomena (coughs, pauses, and so on) corresponding to a

single IF label.

Following the general idea developed in [3] for building the SQL query in an Italian version of ATIS (Air Travel Information Service), the construction of the IF is split in two phases: the argument level is evaluated by looking for particular substrings in the input, while the sequence of speech-act and concept list is built by considering features of the whole segment.

For determining speech act and concepts an approach based on semantic classification trees [5] as described in [3] is used. Its portability is conditioned by the availability of labeled data for the new domain. Actually, since the concatenation of the speech act and the concepts follows a pre-defined syntax, a new approach, still based on semantic classification trees but also exploiting the IF structure, is in progress. Also in this case, the classification module can be built on the basis of the IF syntax and a corpus of annotated dialogues.

For argument extraction, on the other hand, with an approach which is reminiscent of [1], a pipeline of filters is used, where the output of every module is taken as input by the following one. Nevertheless, in the system discussed in this paper, the analysis is semantic rather than syntactic and every filter corresponds to a different IF argument (or to a small set of arguments). Moreover, not all filters are finite-state transducers, because in a few cases a context-free grammar better suits the problem. For this reason, the more general term of filter indicates each step of the pipeline. The filters can be used all together for the most general analysis, or they can be applied separately when only a specific domain is considered.

If the target is a given domain, and filters were developed also for other domains, introducing them in the system could in principle help in disambiguation even when it is used on the single domain. On the other hand, since all the filters are very efficient, the execution time of the whole system would not be significantly affected. This is what results presented in Section 4 suggest, even if it would be necessary to test this hypothesis on a larger number of domains to definitely confirm it.

3.1. Argument extraction in one step

The arguments to be extracted consist of an *argument name* and a *value*. The structure of the values is predefined and includes the possibility of having lists, conjunctions (i.e., AND lists) and disjunctions (i.e., OR lists) of values. This can be nested at more than one level, so that the elements of a list can have their own non atomic structure. The syntax of values is the same for all arguments. On the contrary, for each argument name the IF definition gives the set of all the possible atomic value it can assume. These sets are all finite, with the only exception of some of the values involving numbers.

Each argument usually corresponds to a substring of the input, and substrings corresponding to different arguments do not overlap. Therefore, argument extraction can be performed by identifying the interesting substrings and computing the argument name and the corresponding value. When all argument candidates in the input utterance have been spotted, the consistency of the different values among them and with respect to the domain action needs to be checked. In particular, concepts are important for determining which arguments are legal and which are not.

A unique general semantic grammar can be built for all possible arguments and their values. In fact, this was the approach chosen at the beginning [4], when only the first domain (HORE) was considered. In the choice of the approach, the general idea was to keep all the models as simple as possible, with the future goal of automizing as much as possible its construction. On the other hand, the model could not be too simple, if this caused it to become too large.

Therefore, we decided to use a regular grammar for lexical analysis and an LR(1) grammar for the values computation (with LEX and YACC tools [2]). Using only a regular grammar would be very appealing to us, since automatic learning would be easier. Nevertheless, it was not sufficient for dealing with some of the arguments, especially when there are interruptions, repetitions and/or corrections. Some of these cases are not easily dealt with even by an LR(1) grammar, but their number is quite limited. In any case, such tools are very time efficient and this is always very important in systems where the real-time response is required.

Note that, when dealing with spontaneous speech, it is very important to be able to preserve partial analyses. In fact, each utterance can be suddenly interrupted at every point either because the speaker interrupts his/herself or because the acoustic recognizer introduces some error. In this case, even the translation of part of the argument can be very useful to keep the communication between the two partners. It could happen, for example, that when a date was meant, only the number corresponding to the month day is properly recognized. In cases like that, translating anyway what was got can be very useful, because it can give important cues to the hearer. Even when the hearer is puzzled by the result, he/she can use it as a starting point to ask for clarifications.

This one step approach had advantages and disadvantages. On one side, common partial analyses, such as number interpretation, could be shared. On the other side, nevertheless, it was quite cumbersome to keep control on all possible side effects of each new rule that it became necessary to introduce, either for taking into account new arguments or new realizations of old arguments found in the data.

3.2. Multiple step argument extraction

The porting of the system from the first HORE domain to the TRIN domain (and then to the union of the two) raised the problem of updating the argument extraction. Moreover, the IF definition is not yet stable, and some of the arguments and argument values keep changing. Of course, these changes are normal as the approach is still field of research, but in our opinion, some changes are to

be always expected even for well assessed systems, due to variations in the domain or normal system maintenance. For all these reasons, it was necessary to find out an architecture where changes could be easily done and checked.

In conclusion, the following two properties were the most important to be imposed:

robustness: when an analysis fails, a consistent partial analyses must be produced in any case;

modularity: the architecture must be modular so that the values of each argument can be considered independently from the other arguments; in principle, only necessary arguments could be plugged in when considering a particular domain.

A possibility was to build a different module for each argument, which could be run independently. This ideal architecture was made impossible by mutual relations existing between the different arguments. First of all, there are some expressions, e.g. numbers, which are used by several arguments, like prices, temporal expressions, and also all expressions where a number of objects are indicated ("I'm traveling with four friends", "I need two single rooms", "Is there a meeting room suited for forty three people?"). This means that a few arguments are part of a core which is always present in every possible domain.

Moreover, the structured values can be better built by multiple steps, where the results of the preceding step is used to build next step, whenever this is necessary. In other cases, it is left untouched. For example, numbers are one of the first things to be analyzed. Next steps can consider which of them are better interpreted as hours, or dates, or prices. Nevertheless, some of them are nothing more than numbers and will not be elaborated by any of the following filters.

Therefore, all the filters are aligned in a pipeline, where at each step the input is analyzed and searched for substrings corresponding to a possible value for the argument under consideration. Such substrings can be formed by both words and labels of arguments analyzed in some preceding step. The output will be a transcription of the input where the analyzed substrings are substituted by a new label giving the argument name and its value.

The way in which each filter is implemented obviously depends on the nature of the values to be extracted. Also in this case, the criterion is followed of always trying to use a model as simple as possible. The majority of them will be simple transducers implementing a regular grammar. A more powerful grammar (LR(1)) is used only when this provides effective advantages. Also in these cases, however, the module still appears as a filter as it converts some input substrings in other substrings. The actual implementation required such more complex structures only in about one fourth of the cases. Moreover, all these correspond to very general arguments, i.e. person names, numbers, temporal expressions, distances, prices, while domain specific arguments are generally much simpler. Therefore, in our opinion, the probability of having to develop such a complex filter for a new domain is quite low.

Usually, once a pipeline has been implemented for a non trivial domain, the addition of a new argument only requires the implementation of a new filter, which can be roughly done by only looking at the different values and building for them the corresponding phrases in the considered natural language (Italian, in this case). After that a further step made by using real data usually enhance performance.

Note that there is a core set of mandatory filters which are in charge of preliminary analyses shared by the other filters. For the others, there are some chains for which introducing a certain filter (e.g., complex temporal expressions) implies the introduction of all those that precede it in the chain (e.g., simple time expressions, like dates or hours). Care should be put to avoid unnecessary dependency: no filter should hypothesize the presence of other filters not in the core set. This can usually be managed by introducing more precise rules.

4. EVALUATION

A corpus of person to person monolingual dialogues was collected for both the domains, i.e. HORE and TRIN. During the collection no machine intermediation was used, and the two partners could talk quite freely, with the only constraint of not seeing each other. Moreover, they were asked not to talk at the same time, but this restriction was not always observed.

The two corpora were then divided into a training and a test set. All the data are annotated. In particular, for each semantic segment the list of valid arguments is given, i.e. only the arguments that have to be translated and not the irrelevant ones. The annotation does not bind the arguments to any substring, but only to the complete semantic segment. Only the training set was used for developing the argument extractor. During this process a lot of annotation errors were found and corrected. Obviously this could not be done with the test set.

The evaluation focuses on the argument extraction and in particular on performance variation with respect to the filters introduced in the system. The filters were divided in four groups:

A. the *core* arguments, i.e. all the filters corresponding to analyses which are shared by other filters and which in a way constitute a basis for all the other filters;
B. the *general* arguments, which do not depend on the domain, but on the application; they include temporal and spatial expressions, prices, payment expressions and so on;
C. the *hore* arguments, including all expressions which are typical of the hotel reservation domain;
D. the *trin* arguments, including all expressions which are typical of the transport information domain.

Note that these four groups provide a partition of all the filters, i.e. they are mutually disjoint and their union covers all the filters. Table 1 gives the number of regular rules and of context-free rules.

Table 2 gives performance in terms of precision and recall

		regular	LR(1)
A	core	336	57
B	general	673	143
C	hore	73	-
D	trin	141	-
total		1223	200

Table 1: Total number of regular and context-free rules in each group of filters.

for different subsets of filters. All experiments were performed on the transcriptions of the dialogues, not on the output of the acoustic recognizer. Note anyway that the rules take into account whenever possible the most likely acoustic errors (in particular for short words, especially functional ones).

The row at the top of the table gives the total number of the arguments in the set corresponding to the column. In the rest of the table, percentage precision and recall are given for every group of filters on each of the four sets. The row "A" gives performance of the core filters with respect to the global task: it can be seen that their contribution is very low. Nevertheless, their presence is very important to prepare the following filtering and also in smoothing performance degradation. In the following row, the filters related to the application, but independent from the domain are added to the core ones. Note that using this second set (B) without the first one (A) wouldn't work, because filters in set B need the prefiltering made by A. The successive two rows give performance respectively for a pipeline targeted to the HORE domain and to the TRIN domain. Finally, performance obtained by using all the filters together are given.

	HORE		TRIN	
	Training	Test	Training	Test
# args	4747	1425	6756	1655
	P/R %	P/R %	P/R %	P/R %
A	11.6/12.0	12.2/12.8	6.5/6.6	8.1/8.3
A, B	49.8/52.8	44.4/48.1	44.4/46.7	47.1/49.4
A, B, C	70.5/76.3	67.2/73.9	44.4/46.7	47.1/49.4
A, B, D	49.8/53.2	44.5/48.3	57.8/65.4	60.8/67.8
all	70.3/76.7	67.1/74.2	57.8/65.4	60.7/67.8

Table 2: Precision and recall in argument extraction. The first row (labeled # args) gives the total number of arguments present in the considered set.

5. DISCUSSION AND FUTURE WORK

It has to be noted that the performance on training sets have not been pushed as it could be. In fact, it would have been possible to design special rules for further specific cases. On the other hand, for enlarging the coverage after a certain point it would have been necessary to encode peculiar expressions which seemed very unlikely in a machine mediated conversation or anyway far from standard.

It should be considered, in fact, that data were collected in a direct person-to-person monolingual communication. When the communication is mediated by a machine, on the contrary, the expressions tend to be less imaginative and much more standard. This intuition should be verified on a test set of data collected with the real use of the system or, maybe, by simulating an analogous system with performance a bit better than the real one.

The fact that there is not relevant performance deterioration on the test set with respect to the training set is probably due to this fact. In fact, the more usual expressions have a percentage of occurrences which is very similar in the training and in the test set. As the rules were developed manually, they are likely to be quite general and then to cover these frequent cases quite well.

On the other hand, work is still in progress, especially on complex values and on those arguments that are less crucial for the translation but still would improve its quality.

It is interesting to see that the filters introduced for a certain domain have very slight influence on performance for another domain. It could be expected that since they help to disambiguate cases where interpretation could be difficult, then performance would increase. Anyway, this conclusion can not be definitive, as HORE and TRIN are two very different domains, so that the corresponding arguments can rarely be confused.

6. REFERENCES

[1] S. Abney. Partial Parsing via Finite-State Cascades. In *Proc. of the ESSLLI '96 Robust Parsing Workshop*, 1996.

[2] A. V. Aho, R. Sethi, and J. D. Ullman. *Compilers. Principles, Techniques, and Tools.* Addison-Wesley Publishing Company, Reading, MA, 1985.

[3] M. Cettolo, A. Corazza, and R. De Mori. Language Portability of a Speech Understanding System. *Computer Speech and Language*, 12:1–21, 1998.

[4] M. Cettolo, A. Corazza, F. Pianesi, E. Pianta, and L.M. Tovena. Speech-to-Speech Translation based Interface for Tourism. In *ENTER99*, Innsbruck, Austria, 1999.

[5] R. De Mori and R. Kuhn. The Application of Semantic Classification Trees to Natural Language Understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(5):449–460, May 1995.

[6] L. Levin, D. Gates, A. Lavie, and A. Waibel. An Interlingua Based on Domain Actions for Machine Translation of Task-Oriented Dialogues. In *Proceeding of the International Conference on Spoken Language Processing*, Sydney, Australia, 1998.

[7] M.Cettolo and D.Falavigna. Automatic Detection of Semantic Boundaries based on Acoustic and Lexical Knowledge. In *Proceeding of the International Conference on Spoken Language Processing*, Sidney, Australia, 1998.

[8] A. Waibel. Interactive Translation of Conversational Speech. *Computer*, 29(7):41–48, 1996.