Development of a stochastic dialog manager driven by semantics

Francisco Torres, Emilio Sanchis and Encarna Segarra

Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia, Spain
{ftgoterr, esanchis, esegarra}@dsic.upv.es

10.21437/Eurospeech.2003-257

Abstract

We present an approach for the development of a dialog manager based on stochastic models for the representation of the dialogue structure and strategy. This dialog manager processes semantic representations and, when it is integrated with our understanding and answer generation modules, it performs natural language dialogs. It has been applied to a Spanish dialogue system which answers telephone queries about train timetables.

1. Introduction

The development of spoken dialogue systems in semantically restricted domains, such as natural language queries to database systems, is a field of special interest in Speech Technologies. The main objective of such systems is to make it posssible for the human user to query an information system allowing for a mixed initiative which is natural and spontaneous.

These systems are characterised by the following: telephone access, which is practical, tasks restricted to specific domain which limits the vocabulary, and mixed initiative. Descriptions of some of these systems which have been developed in the last years can be found in [1], [2], [3], [4], [5] y [6].

Our work has been developed within the framework of the BASURDE Spanish project [7], whose aim is the construction of a spoken dialog system by telephone access to a train timetable information system. To control this system, we propose a dialog manager which processes semantic representations of user turns and represents the structure and strategy of the dialogs through stochastic models, which are learnt from a corpus labelled in terms of dialog acts or frames. A preliminary version of this dialog manager can be found in [8].

In this task, sparse training data leads to the problem of insufficient estimation of the models. To solve this, we have developed methods of generalization and smoothing. One method is the use of two dialog models, each of which represents a different level of knowledge about the task. The other method is the generalization of the semantic representations of the user turns. Both methods increase the number of available alternatives in any dialog turn, giving the dialog manager more alternatives for driving the dialog.

In this work, we present our dialog manager. This dialog manager receives and produces semantic representations. It has been integrated with the understanding module [9], the reply generation module and the database manager, so that the whole system receives and produces Spanish sentences. We also present an evaluation of the whole system.

2. Semantic representation

As we have already stated, our system integrates four modules: an understanding module, a dialog manager, a database query module and a reply generator. The input and the output of the whole system are written Spanish sentences in the domain of the task.

The understanding module translates user natural language sentences into a set of frames which represent the meaning of each user turn. The input to the dialog manager are these frames or semantic representations of user turns. Figure 1 shows an example of a user turn and its semantic representation.

Once the dialog manager has chosen how the system will reply to the user, it builds another set of frames. These are supplied to the next module, the reply generator, which does the inverse translation: from system semantic representation to system answer in natural language.

U0: Hola, quería los horarios de trenes para ir de Burgos a Bilbao el dos de Marzo por la mañana / *Hello, I want the train timetables from Burgos to Bilbao for March the second in the morning.*
U0: (DEPART-HOUR)
ORIGIN: Burgos
DESTINATION: Bilbao
DEPART-DATE: 03-02-2003
INTERVAL DEPART-HOUR: 05.00-13.00

Figure 1: *Semantic representation example*

3. The estimation of the dialog model

The task defined in the BASURDE project consists of telephone queries about timetables, prices and services of long distance Spanish trains. The corpus contains 215 dialogs which were acquired using the Wizard of Oz technique. This corpus has 1,460 user turns (14,902 words).

The structures of the dialog model are represented by dialog acts. We have proposed a set of dialog acts with three levels of labelling [10]. The first level describes the general acts of any dialog, independently of the task, such as openings, queries, confirmations, answers, etc. The second level is related to the semantic representation of the turn and is specific to the task (departure time, price, train type...). The third level represents the values of the attributes given in the turn. Each turn in the corpus is labelled with the identifiers of one or several dialog acts according to this proposal.

The stochastic dialog models have been estimated from the first and second levels only. In order to avoid the limitations of training with a small corpus, we have built two

models, a specific one and a general one. The specific model has 310 states. Each state is identified by a concatenation of one or more strings *(Turn:FirstLevel:SecondLevel)*, where *Turn* identifies the user or the system, *FirstLevel* identifies the dialog act and *SecondLevel* identifies the attributes included in that dialog act. The general model has 74 states identified by a concatenation of one or more strings *(Turn:FirstLevel)* and, thus, represents a more general knowledge level. From now on, we refer to the set of two dialog models as *DM*, to the specific model as *L2_Model* and to the general model as *L1_Model*.

4. Dialog manager overview

The behaviour of the dialog between user and system depends on two components, the stochastic dialog model (*DM*) and the historic register (*HR*), which is updated by the dialog manager.

Our stochastic models are bigram models and, therefore, the set of dialog acts for a turn is predicted from the set of dialog acts from the previous turn. In some situations the model may not have an answer due to understimation caused by lack of training data. In order to overcome this problem, we have implemented two stochastic models, as well as a generalization process for the input frames.

A bigram model is blind to all the information shared between the system and the user during the previous turns. Obviously, the normal development of any dialog implies the knowledge of data interchanged at any given moment. Therefore, the system needs to memorise all relevant information from the beginning of the dialog. In our dialog management module, the HR performs this function.

The dialog manager uses these two complementary sources of information, *DM* and *HR*, in order to decide the system reply. The algorithm essentially consists of an iteration in which the two components are updated by user and system turns until the dialog ends. Figure 2 shows the dialog manager algorithm.

```
Initialisation (HR);          /* Historic Register */
Read (DM);                    /* Stochastic Dialog Models */
DM_state = Opening;           /* Initial state of dialog */
Repeat
      /* reading user frames and semantic generalization */
  Read (frame_input);
  DM_input = Adapt (HR, frame_input);
      /* model update and register update by user turn */
  DM_state = Update (DM_state, DM_input);
  HR = Update (HR, frame_input);
      /* model update and register update by system turn */
  DM_state = Update (DM_state, HR);
  HR = Update (HR, DM_state, DB_output);
      /* building and writing system frames */
  frame_output = Adapt (DM_state, HR);
  Write (frame_output);
Until      DM_state = Closing
```

Figure 2: *Dialog manager algorithm*

5. Semantic generalization in the DM

As we have already stated, for the *L2_Model* any user semantic representation (or *frame_input*) is a concatenation of one or more strings *(Turn:FirstLevel:SecondLevel)*. The system could attempt to update the state of the DM with the semantic representation: *DM_state = Update (DM_state, frame_input)*. However, unacceptable situations sometimes occur and the system locks-out due to absence of a valid transition from the state of the DM.

To avoid this problem, we perform a preprocessing of the user frames, building a semantic generalization: *DM_input = Adapt (HR, frame_input)* in order to provide more flexible input to the model. The semantic generalization generates a set of slight formal modifications in the user frames. The generated strings, *DM_input*, can differ more or less in form, but must maintain the content of the user frames as much as possible.

Afterwards, we use the set of semantic generalizations to update the model: *DM_state = Update (DM_state, DM_input)*. Thus, we search for a valid transition not only by matching a specific string (*frame_input*), but by matching any string in the set (*DM_input*). The capacity to find transitions increases with the number of strings available for searching transitions.

In the current version of our dialog manager, we apply three methods of generalization of the user semantic representation. Figure 3 shows an example of how these methods operate.

Given 3 dialog acts: *frame_input = (U:Affirmation:Destination,Train_type)(U:Question:Depart_hour,Price)(U:Question:Return_depart_hour)*

Examples of the methods of semantic generalization:

DM_input by unconditional concatenation:
(U:Question:Depart_hour,Price)(U:Question:Return_depart_hour)

DM_input by integration:
(U:Question:Depart_hour,Price,Return_depart_hour)

DM_input by random fragmentation:
(U:Affirmation:Train_type)(U:Question:Price)

Figure 3: *Examples of semantic generalization*

The unconditional concatenation method builds new inputs for updating dialog model by pruning the oldest frames in the input sequence. The underlying idea is that the input frames can be interpreted as a chronological list whose first items have less importance than the last ones.

The integration method is applied when two or more dialog acts appear in the input frames, as for example *(Turn:Act-1:Attrib-11,...,Attrib-1N) (Turn:Act-2:Attrib-21,..., Attrib-2N)*, and *Act-1=Act-2*. These dialog acts can be replaced in the new input by a string whose *SecondLevel* identifier concatenates all the attributes in them: *(Turn:Act-1:Attrib-11,...,Attrib-1N,Attrib-21,...,Attrib-2N)*. This method do not lose any content from the original input.

The random fragmentation method is just the reverse operation of the integration method. When a dialog act includes a rather long list of attributes, the model transition

probability decreases. The method randomly chooses a subset of the attributes and builds a new input with a shorter list of attributes. As in the concatenation method, some content of the original input is lost.

6. The use of the two stochastic dialog models

In order to carry out the *L2_Model* update, we have to build the semantic generalization set, to establish priorities in this set, and to relax the matching criteria, when it is needed. The goodness of this procedure has been proved by its very low failure rate. However, this is sometimes not enough to assure successful updating. In these special situations, the dialog manager uses the *L1_Model* to find a suitable transition.

If the dialog manager achieves only the *L1_Model* update, its current state, say *L1_state*, allows the system to take the *L2_Model* to a new state, say *L2_state*, from which the dialog will follow. This special procedure, first extracts the dialog act labels from the *L1_state* and then searches all the *L2_Model*'s states that include those dialog act labels. From this subset, the one with the highest probability is chosen as the new *L2_state*.

To summarize, both models are updated during the execution of the dialog manager. The dialog manager always builds its replies to the user from the *L2_Model* state, but, it sometimes can not find a transition in the *L2_Model* and so it uses the *L1_Model* state to force a reasonable update of the main model.

7. The historic register

The historic register (*HR*) is a memory where the system stores information about the dialog from its beginning. Thus, the HR complements the current model state, *DM_state*, which only represents the dialog acts of the previous turn.

During the dialog, the system updates the HR. The new values of the attributes in the user frames are stored in the HR together with a reference to the turn number. The system takes this information into account in order to coherently update the dialog model. The system can also modify the HR when it performs a database query. The information returned from the database manager will be also stored in the HR, as we discuss below.

Sometimes, the dialog manager also needs to use the HR in order to generate certain semantic generalizations. For example, when there is an ellipsis, the user can confirm or refuse certain attributes, but does not specify them because it was already done in a previous turn. The system has to know the implicit attributes and only the HR can provide that information. Figure 4 shows an example of a dialog fragment with an ellipsis.

Another type of ellipsis occurs when the user explicitly changes the value of certain attribute, say *A*, and this causes the implicit change of another attribute, say *B*, because there is a logical dependence $A \rightarrow B$. The system must know all the logical relations between attributes and, through the HR, can also take into account these implicit values. A situation which is specially relevant for this task occurs when the user explicitly interchanges origin and destination cities and the system, by means of the HR, detects the implicit change between the trip to the destination and the return trip.

U1: Quería saber los horarios del cuatro de marzo, de Burgos a Bilbao. / *I want to know the train timetables from Burgos to Bilbao for March the fourth.*
U1: (DEPART-HOUR)
[ORIGIN: Burgos] [DESTINATION: Bilbao]
[DEPART-DATE: 04-03-2003]
S1: (WAIT)
(INFORMATION)
[NUMBER-OF-TRAINS: 3] [DEPART-HOUR: 17.00]
(NEW-QUERY)
S1: Un momento, por favor. Hay tres trenes. El primer tren sale a las cinco. ¿Quiere hacer otra consulta? / *One moment, please. There are three trains. The first train leaves at five o'clock. Do you want anything else?*
U2: Sí. ¿Y el siguiente? / *Yes. ¿And the next?*
U2: (AFFIRMATION)
( ) [RELATIVE-ORDER: next]
S2: (INFORMATION) [DEPART-HOUR: 19.00]
(NEW-QUERY)
S2: Sale a las siete. ¿Quiere hacer otra consulta? / *It leaves at seven o'clock. Do you want anything else?*

Figure 4: *Dialog fragment*

Once the system has gathered enough information about any user request, a database query is executed. The records returned from the database manager will usually contain more information than necessary for the immediate system reply. However, in the following turns, the user may ask for more information given by the same query.

Thus, we decided to store all these data by extending the HR to a two level structure. The basic historic register (BHR) contains only the current values of all the attributes together with some control information (times of last update, times of last reference...). The extended historic register (EHR) stores all the information returned from all the queries executed from the beginning of the dialog.

Any query is defined by the values of the three attributes: origin city, destination city and departure date. The function that determines whether a new query to the database is needed searches for coincidence in these values between BHR and EHR. Thus, the number of database queries is minimized and the wait times in the dialog manager are reduced.

8. Experimental results and some conclusions

In order to evaluate the dialog manager, we have chosen thirty scenes and we have experimented with the whole system (the understanding module, the dialog manager and the reply generator). We measure the goodness of the dialog manager by the level of quality of the Spanish sentences generated.

The reply generator translates system turn semantic representations (the output of the dialog manager module) into Spanish sentences, according to a set of templates and certain combination rules between these templates and the attribute values. It is mandatory for certain attributes to be included in the system reply. For instance, when the reply is the result of a database query, the attribute values extracted from the database must be included in the reply. However, other attributes could be included as a kind of implicit

confirmation, specially those which have changed their values in the previous user turn.

We have considered three types of scenes in the experimentation. In type A, the user asks for train timetables for a one-way itinerary. In type B, the user asks for train timetables and prices for a one-way itinerary. Finally, in the type C, the user asks for train timetables and prices for a round-trip itinerary. For each dialog we have annotated the rate of success in answering the query, the number of turns required, the number of explicit confirmations that the system needed and the number of corrections that the user had to make in order to redirect the dialog. The following table shows the average values of these parameters for each scene type and the summary.

Table 1: Results of the evaluation

	A	B	C	Total
Success rate	90%	80%	70%	80%
Nº turns	5.0	6.6	8.8	6.8
Nº confirmations	1.1	1.1	1.2	1.1
Nº corrections	0.4	0.7	1.3	0.8

We can conclude that the stochastic modelization of the dialog behaviour is an acceptable approach for the design of a Dialog Manager. Our proposal of generalization of the input frames and the use of the two stochastic models combined with the historic register allows the system to deal with many situations not seen in the training corpus. In this way, we can tackle the problem of the lack of training data (only 215 dialogs in the BASURDE task) in order to automatically learn a feasible dialog system.

Nowadays, we are working on the incorporation of confidence measures into the understanding module and into the dialog manager. In this way, we expect to achieve an acceptable performance of the dialog system, when spontaneous speech will be used.

9. Acknowledgement

This work has been partially funded by CICYT under project TIC2002-04103-C03-03, Spain.

10. References

[1] Cmu communicator spoken dialog toolkit (csdtk). http://www.speech.cs.cmu.edu/communicator/.

[2] Pieraccini, R. y Levin, E., "AMICA: the AT&T Mixed Innitiative Conversational Architecture", *European Conference on Speech Communication and Technology EUROSPEECH*, 1875-1878, Rhodes (Grecia), 1997.

[3] Lamel, L., Rosset, S., Gauvain, J. L., Bennacef, S., Garnier-Rizet, M. y Prouts, B., "The LIMSI ARISE System", *Speech Communication*, 31(4):339-353, 2000.

[4] Glass, J. y Weinstein, E., "Speech builder: facilitating spoken dialog system development", *European Conference on Speech Communication and Technology EUROSPEECH*, 1335-1338, Aalborg (Dinamarca), 2001.

[5] Córdoba, R., San-Segundo, R., Montero, J.M., Colás, J., Ferreiros, J. Macias-Guarasa, J. y Pardo, J.M., "An interactive directory assistance service for Spanish with large vocabulary recognition", *European Conference on Speech Communication and Technology EUROSPEECH*, 1279-1282, Aalborg (Dinamarca), 2001.

[6] López-Cozar, R., Rubio, A. J., García, P., Díaz-Verdejo, J. E. y López-Soler, J. M., "Sistema telefónico de información a viajeros", *Primeras Jornadas en Tecnología del Habla*, Sevilla (España), 2000.

[7] Bonafonte, A., Aibar, P., Castell, N. Lleida, E., Mariño, J.B., Sanchis, E. y Torres, I., "Desarrollo de un sistema de diálogo oral en dominios restringidos", *Primeras Jornadas en Tecnología del Habla,* Sevilla (España), 2000.

[8] Torres, F., Sanchis, E. y Segarra, E., "Desarrollo de un gestor de diálogo basado en modelos estocásticos y dirigido por la semántica", *Procesamiento del Lenguaje Natural*, 29, 175-180, 2002.

[9] Segarra, E., Sanchis, E., Galiano, M., García, F. y Hurtado, L., "Extracting Semantic Information through Automatic Learning Techniques", *International Journal of Pattern Recognition and Artificial Intelligence*, 16(3), 301-307, 2001.

[10] Martínez, C., Sanchis, E., García, F. y Aibar, P., "A labeling proposal to annotate dialogues", Third International *Conference on Language Resources and Evaluation, LREC*, 1577-1582, Las Palmas (España), 2002.