A framework to develop context-aware adaptive dialogue systems

David Griol¹, Zoraida Callejas², Ramón López-Cózar²

¹Dept. of Computer Science, Carlos III University of Madrid, Spain
²Dept. of Languages and Computer Systems, University of Granada, Spain
dgriol@inf.uc3m.es, {zoraida,rilopezc}@ugr.es

Abstract

In this paper, we propose a general-purpose framework to develop spoken dialogue systems that dynamically adapt their behavior to user requirements and preferences, as well as to the interaction context. A data-driven technique is proposed to build task structures and dialogue models within the framework. Our proposal reduces the effort required for both the implementation of a new system and the adaptation of an existing one to a new task. We have evaluated the framework developing a travel-planning system, and provide a detailed discussion of its positive influence on both the interaction quality and the personalization of provided services.


1. Introduction

In human conversation, speakers adapt their message and the way they convey it to their interlocutors and to the context in which the dialogue takes place. Thus, the interest in developing dialogue systems capable of maintaining a conversation as natural and rich as a human conversation has fostered research on adaptation of these systems considering contextual information. Because of this reason, dialogue systems are required to recognize the context of the interaction and act accordingly.

Adaptation can play a relevant role in speech-based applications [1, 2, 3]. For example, users have diverse ways of communication. Novice users and experienced users may want the interface to behave completely differently, such as maintaining more guided vs. more flexible dialogues. In these cases, processing context is not only useful to adapt the systems’ behaviour, but also to cope with the ambiguities derived from the use of natural language [4, 5, 6]. For instance, contextual information can be used to resolve anaphoric references depending on the context of the dialogue or the user location.

The performance of a dialogue system also depends highly on its ability to adapt to the environmental conditions, such as other people speaking near the system or noise generated by other devices. Thus, it is necessary to provide an effective and dynamically adapted interaction between users and systems [7, 8, 9]. In addition, although much work emphasizes the importance of taking into account contextual information not only to solve the tasks presented to the dialogue system by the user, but also to enhance the system performance in the communication task, this information is not usually considered when designing a dialogue model [4, 10]. For these reasons, there has been a growing interest during the last decade in developing statistical approaches to model the different modules that compose a dialogue system, as well as the user interaction with the system [11, 12].

In this paper, we propose a framework for the development of spoken dialogue systems that provides them with context-aware adapted services accessible through natural language. To facilitate general-purpose behaviour and reduce the effort required for both the implementation of a new system and the adaptation of an existing one to a new task, we propose to use a statistical methodology that dynamically combines dialogue management and user’s intention modelling to improve the selection of system responses by taking into account both the internal and external context of the interaction. We have developed a system following the proposed framework and compared it with a baseline. Experimental results show that the proposed framework improves system performance as well as the user perceived quality.

2. The proposed framework

Figure 1 shows the architecture of the proposed framework for developing context-aware spoken dialogue systems. As can be observed, a number of systems and modules cooperate to provide adapted information and services. The Facilitator system supplies different services to the dialogue system. Context information is acquired and managed by means of the Positioning system and the Log Analyser system. The former communicates with the ARUBA positioning system to extract and transmit positioning information of the user, whereas the latter generates and updates the user profile. These two systems were developed using the Appear IQ Platform in a previous study [13].

To deal with contextual information and personalize the provided services, we propose to incorporate two new modules in the dialogue system architecture as shown in Figure 1. The first module, called Context Manager, deals with context information associated with users and the interaction context. The second module, called User’s intention recogniser, helps the dialogue manager to detect and anticipate the user’s intention.

In order to control the interactions with the user, our dialogue manager represents dialogues as a sequence of pairs \((A_i, U_i)\), where \(A_i\) is the output of the dialogue system (the system answer) at time \(i\), and \(U_i\) is the semantic representation of the user turn (the result of the understanding of the user input) at time \(i\); both expressed in terms of dialogue acts [14]. This way, each dialogue is represented by:

\[
(A_1, U_1), \ldots, (A_i, U_i), \ldots, (A_n, U_n)
\]

where \(A_1\) is the greeting turn of the system, and \(U_n\) is the last user turn. We refer to a pair \((A_i, U_i)\) as \(S_i\), the state of the dialogue sequence at time \(i\).

We consider that at time \(i\) the objective of the dialogue manager is to find the best system answer \(A_i\). Then, the objective of the user intention recogniser at time \(i\) is to detect the user
intention $UI_i$ for the current state of the dialogue. This selection is a local process for each time $i$ and takes into account the previous history of the dialogue, i.e., the sequence of states of the dialogue preceding time $i$:

$$\hat{A}_i = \arg\max_{A_i \in A} P(A_i | S_1, \ldots, S_{i-1}) \quad (1)$$

$$\hat{UI}_i = \arg\max_{UI_i \in U} P(UI_i | S_1, \ldots, S_{i-1}, A_i) \quad (2)$$

where the sets $A$ and $U$ contain all the possible system and user answers respectively.

As the number of possible sequences of states is very large, we establish a partition in this space (i.e., in the history of the dialogue up to time $i$) by defining two data structures. Let $DR_i$ be what we call Dialogue Register at time $i$. The dialogue register can be defined as a data structure that contains information about concepts and values of attributes provided by the user throughout the previous dialogue history and/or the external context provided by the context manager (e.g., the user’s current position). Let $UR_i$ be what we call User Register at time $i$. The user register can be defined as a data structure that contains the dialogue acts provided by the user utterances during the dialogue and the information in the user profile.

The user profile is comprised of user’s: i) Id, which he can use to log in to the system; ii) Gender; iii) Experience, which can be either 0 for novel users (first time the user calls the system) or the number of times the user has interacted with the system; iv) Skill level, estimated taking into account the level of expertise, the duration of their previous dialogues and the time that was necessary to access a specific content and the date of the last interaction with the system. A low, medium, high or expert level is assigned using these measures; v) Most frequent objective of the user; vi) Reference to the location of all the information regarding the previous interactions and the corresponding objective and subjective parameters for the user.

After applying the above considerations and establishing the equivalence relations in the histories of dialogues, the selection of the best $A_i$ and $UI_i$ is given by:

$$\hat{A}_i = \arg\max_{A_i \in A} P(A_i | DR_{i-1}, S_{i-1}) \quad (3)$$

$$\hat{UI}_i = \arg\max_{UI_i \in U} P(UI_i | UR_{i-1}, A_i) \quad (4)$$

Figure 2 describes the method that we propose to adapt the dialogue manager by taking into account both the internal and the external context. As can be observed, the output generated by the user intention recogniser is taken into account by the dialogue manager to select the next system response. This way, the selection can be modelled by the following equation:

$$\hat{A}_i = \arg\max_{A_i \in A} P(A_i | DR_{i-1}, S_{i-1}, \hat{UI}_{i-1}) \quad (5)$$
term \( (DR_{i-1}, S_{i-1}, U_i) \). The values of the output layer can be viewed as the a posteriori probability of selecting the different system responses given the current situation of the dialogue.

### 3. Practical application

We have applied the proposed context-aware framework to develop an adaptive spoken dialogue system for the travel-planning domain. The system provides context-aware information in natural language in Spanish about transportation means to get to a city, flight schedules, weather forecast, car rental, hotel booking, tourist attractions, theatre listings, and film showtimes. The interaction is adapted to take the user context into account.

Using the `City_Transportation` functionality, it is possible to know how to get to a specific city using different means of transport. Contextual information is taken into account to adapt this information considering the user’s current position, preferred means of transportation and city. The `Flight_Schedules` functionality provides flight information considering the user’s requirements. Users can provide the origin and destination cities, ticket class, departure and/or arrival dates, and departure and/or arrival hours.

Using `Weather_Forecast` it is possible to obtain the forecast for the required city and dates (for a maximum of 5 days from the current date). The contextual information taken into account includes the user’s current location, preferred dates and/or hours, and preferred ticket class. The `Car_Rental` functionality provides this information taking into account the users’ requisites including the city, pick-up and drop-off date, car type, company, driver’s age, and office. The `Hotel_Booking` functionality provides hotels which fulfil the user’s requirements (city, name, category, check-in and check-out dates, number of rooms, number of guests).

The `Tourist-Attractions` functionality provides information about places of interest for a specific city. This information is mainly based on the user’s recommendations. The `Theatre_Listings` and `Film_Showtimes` respectively provide information about theatre performances and film showtimes that takes into account the user’s requirements (e.g., city, name of the theatre/cinema, name of the show/film, category, date, and hour).

The semantics of utterances is modelled using frames [15]. We defined eight concepts to represent the different queries that the user can perform (`City_Transportation`, `Flight_Schedules`, `Weather_Forecast`, `Car_Rental`, and `Hotel_Booking`, `Tourist-Attractions`, `Heater-Listings`, and `Film-Showtimes`). Three task-independent concepts have also been defined for the task (Affirmation, Negation, and Not-Understood). A total of 101 system actions (DAs) were defined taking into account the information that the system provides, requests or confirms.

The `DR` defined for the task is a sequence of 57 fields, corresponding to: i) The 8 concepts defined for the dialogue act representation (`City_Transportation`, `Flight_Schedules`, `Weather_Forecast`, `Car_Rental`, and `Hotel_Booking`, `Tourist-Attractions`, `Theatre_Listings`, and `Film_Showtimes`); ii) A total of 45 possible attributes for the concepts (defined in Table 1); iii) The 3 task-independent concepts that users can provide (`Acceptance`, `Rejection` and `Not-Understood`); iv) A reference to the user profile.

A set of 150 scenarios were manually defined to consider the different queries to the system including different user requirements and profiles. Basic scenarios defined only one objective; i.e. the user aims at obtaining information about only one type of the possible queries to the system (e.g., flight schedules from a city to another for a specific date). More complex scenarios included more than one objective for the dialogue (e.g., to obtain information about how to get to a specific city, as well as car rental and hotel booking information). For each scenario, once the contextual information was received by the Context Manager, it loaded the specific context profile characteristics. This information was then checked by the rest of modules in the dialogue system to personalize the provided service. An example of complex scenario is as follows:

<table>
<thead>
<tr>
<th>Query</th>
<th>Semantic attributes</th>
<th>System responses</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>City_Transportation</code></td>
<td><code>City, Means_Transport, Origin_City</code></td>
<td>Ask and Confirm each attribute, Provide City_Transportation</td>
</tr>
<tr>
<td><code>Flight_Schedules</code></td>
<td><code>Origin City, Destination City, Departure Date, Departure Hour, Arrival Date, Arrival Hour, Ticket Class</code></td>
<td>Ask and Confirm each attribute, Provide Flight_Schedules</td>
</tr>
<tr>
<td><code>Weather_Forecast</code></td>
<td><code>City, Country, Date</code></td>
<td>Ask and Confirm each attribute, Provide Weather_Forecast</td>
</tr>
<tr>
<td><code>Car_Rental</code></td>
<td><code>City, Country, Pick_Up_Date, Drop_Off_Date, Car_Type, Company, Driver_Age, Office</code></td>
<td>Ask and Confirm each attribute, Provide Car_Rental</td>
</tr>
<tr>
<td><code>Hotel_Booking</code></td>
<td><code>City, Country, Hotel_Name, Hotel_Category, Check_in_Date, Check_out_Date, Number_Rooms, Number_People</code></td>
<td>Ask and Confirm each attribute, Provide Hotel_Booking</td>
</tr>
<tr>
<td><code>Tourist-Attractions</code></td>
<td><code>Country, City</code></td>
<td>Ask and Confirm each attribute, Provide Tourist-Attractions</td>
</tr>
<tr>
<td><code>Theatre_Listings</code></td>
<td><code>Country, Category, Show, Theatre, Date, Hour</code></td>
<td>Ask and Confirm each attribute, Provide Theatre_Listings</td>
</tr>
<tr>
<td><code>Film_Showtimes</code></td>
<td><code>Country, Category, Film, Cinema, Date, Hour</code></td>
<td>Ask and Confirm each attribute, Provide Film_Showtimes</td>
</tr>
</tbody>
</table>

Table 1: Semantic representation for the travel-planning domain

We evaluated the behaviour of the system with real users and compared its performance with that of a non-context-aware version of the system. A total of 150 dialogues were recorded from interactions of six users employing the context-aware and the non context-aware system. The evaluation was carried out by students and lecturers in our department following the types of scenario described above. These users were in different settings and used their own devices. Objective and subjective evaluations were carried out. We considered the following measures for the objective evaluation: i) Dialogue success rate (Success); ii) Average number of turns per dialogue (ntT); iii) Confirmation...
tion rate (Confirmation); and iv) Error correction rate (ECR). The confirmation rate was computed as the ratio between the number of explicit confirmation turns and the total number of turns in the dialogue. The ECR was computed as the number of errors detected and corrected by the dialogue manager divided by the total number of errors.

The results presented in Table 2 show that both systems interacted correctly with the users in most cases. However, the context-aware system obtained a higher success rate, improving the results of the non context-aware system by 12% absolute. Using the context-aware system, the average number of required turns was also reduced from 15.6 to 8.4.

The confirmation and error correction rates were also improved by the context-aware system, given that less information was required from the user, reducing the probability of causing ASR errors. The main problem was related to user inputs misrecognized with very high ASR confidence, which was forwarded to the dialogue manager in both systems. However, as the success rate shows, this fact did not have a considerable impact on the system operation.

<table>
<thead>
<tr>
<th>Dialogue features</th>
<th>Non Context-Aware</th>
<th>Context-Aware</th>
</tr>
</thead>
<tbody>
<tr>
<td>Success</td>
<td>94.0%</td>
<td>99.5%</td>
</tr>
<tr>
<td>n/t</td>
<td>8.4</td>
<td>2.0</td>
</tr>
<tr>
<td>Confirmation</td>
<td>26%</td>
<td>26%</td>
</tr>
<tr>
<td>ECR</td>
<td>0.98%</td>
<td>0.29%</td>
</tr>
</tbody>
</table>

Table 2: Results of the objective evaluation

In addition, we asked the users to fill-in a questionnaire to assess their subjective opinion about the system performance. The questionnaire had five questions: i) Q1: How well did the system understand you?; ii) Q2: How well did you understand the system messages?; iii) Q3: Was it easy for you to get the requested information?; iv) Q4: Was the interaction rate adequate?; v) Q5: Was it easy for you to correct the system errors?. The possible answers for the questions were the same: Never, Seldom, Sometimes, Usually, and Always. All the answers were assigned a numeric value between one and five (in the same order as they appear in the questionnaire). Table 3 shows the average results of the evaluation.

<table>
<thead>
<tr>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non Context-Aware</td>
<td>4.1</td>
<td>4.5</td>
<td>3.9</td>
<td>3.6</td>
</tr>
<tr>
<td>Context-Aware</td>
<td>4.3</td>
<td>4.1</td>
<td>4.2</td>
<td>3.8</td>
</tr>
</tbody>
</table>

Table 3: Results of the subjective evaluation (1=worst, 5=best)

From the results set out in both tables, it can be observed that both systems were considered to correctly understand the different user queries and obtained a similar evaluation regarding the facility of correcting errors caused by the ASR module. However, the context-aware system had a higher evaluation rate regarding the facility of obtaining the data required to fulfil the objectives in the scenarios, and regarding the suitability of the interaction during the dialogue.

By means of high-level dialogue features, we evaluated the duration of the dialogues, how much information was transmitted in single turns, and how active the dialogue participants were. These dialogue features were considered by the following statistical properties: i) Different dialogues: percentage of different dialogues with respect to the total number of dialogues, and number of repetitions of the most observed dialogue; ii) Turn length: average number of actions per turn; iii) Participant activity: number of user turns in the most observed, shortest and longest dialogues. Table 4 shows the comparison of the high-level measures for the context-aware and non context-aware system.

<table>
<thead>
<tr>
<th>High-level groups</th>
<th>Dialogue features</th>
<th>Non Context-Aware</th>
<th>Context-Aware</th>
</tr>
</thead>
<tbody>
<tr>
<td>Different dialogues</td>
<td>Percentage of different dialogues</td>
<td>85.5%</td>
<td>71.7%</td>
</tr>
<tr>
<td>Repetitions of the most observed dialogue</td>
<td>15</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Turn length</td>
<td>Average number of actions per turn</td>
<td>1.7</td>
<td>1.4</td>
</tr>
</tbody>
</table>

Table 4: Comparison of results regarding the high-level dialogue features

As can be observed, there is a reduction in the number of turns of the longest, shortest and most frequent dialogues for the context-aware system. The number of different dialogues is also lower using the context-aware system due to the reduction in the number of turns, which can be observed in the number of repetitions of the most frequent dialogue. This happened because users had more variability in order to provide the information needed to access the services in the non context-aware system.

Finally, we evaluated both systems considering the frequency of user and system dialogue acts, as well as the proportion of goal-directed dialogue acts, versus the grounding dialogue acts to confirm data. We noticed that in both cases there were significant differences in the distribution of dialogue acts.

On the one hand, users needed to provide less information using the context-aware system. This explains the higher proportion for the rest of user actions in the context-aware system. We also observed a higher proportion of yes/no actions for the context-aware dialogues, which were mainly used to confirm that specific services had been correctly provided using the contextual information. On the other hand, there was a reduction in system requests when the context-aware system was used. This explains the higher proportion in the inform and confirmation system actions in the context-aware system.

5. Conclusions

Context-aware systems along with mobile devices offer new opportunities in the areas of knowledge representation, natural language processing and intelligent information retrieval from the web. In this paper, we have combined different aspects from these important research areas to provide context aware adaptable web information and services by means of speech interaction, which facilitates personalized and more natural access to information. In order to do this, we have contributed a framework, which can be used to develop context-aware spoken dialogue systems that can be easily integrated in hand-held mobile devices. The framework is comprised of an architecture in which different systems and modules cooperate to provide adapted services, and a representation model for knowledge sharing between the components of the architecture. In future work we will carry out a detailed study on user rejections of the system-hypothesized actions by using the values extracted from the user profile. Also, we will study the benefits that could be derived from including additional interaction modalities.

6. Acknowledgements

Research funded by the Spanish project ASIES TIN2010-17344.
7. References


