Soft Computing

Improving error detection and correction in spoken dialog systems

--Manuscript Draft--

Manuscript Number:	SOCO-D-16-00611
Full Title:	Improving error detection and correction in spoken dialog systems
Article Type:	S.I. : SOCO15
Keywords:	Error Detection and Correction; User Modeling; Spoken Dialog Systems; Human Machine Interfaces; Spoken Interaction
Corresponding Author:	David Griol Universidad Carlos III de Madrid SPAIN
Corresponding Author Secondary Information:	
Corresponding Author's Institution:	Universidad Carlos III de Madrid
First Author:	David Griol
First Author Secondary Information:	
Order of Authors:	David Griol
	Jose Manuel Molina
Funding Information:	
Abstract:	One of the main problems which must be considered during the interaction with a spoken conversational interface is the propagation of errors through the different modules in the system. In this paper, we propose a novel framework to improve the error detection and correction processes in spoken conversational interfaces. The framework combines user behavior and error modeling to estimate the probability of the presence of errors in the user utterance. This estimation is considered as an additional input of the dialog manager and used to compute whether it is necessary to correct possible errors. We have designed an strategy differentiating between the main misunderstanding and non-understanding scenarios, so that the dialog manager can provide an acceptable tailored response when entering the error correction state. We also contribute the practical application of the proposed technique for a dialog system acting as a customer support service.Our results show the appropriateness of our technique to correctly detect and react to errors, enhancing the system performance and user satisfaction.
Section/Category:	Methodologies & Application

Noname manuscript No.
(will be inserted by the editor)

A framework for improving error detection and correction in spoken dialog systems

David Griol · José Manuel Molina

Received: date / Accepted: date

Abstract One of the main problems which must be considered during the interaction with a spoken conversational interface is the propagation of errors through the different modules in the system. In this paper, we propose a novel framework to improve the error detection and correction processes in spoken conversational interfaces. The framework combines user behavior and error modeling to estimate the probability of the presence of errors in the user utterance. This estimation is considered as an additional input of the dialog manager and used to compute whether it is necessary to correct possible errors. We have designed an strategy differentiating between the main misunderstanding and non-understanding scenarios, so that the dialog manager can provide an acceptable tailored response when entering the error correction state. We also contribute the practical application of the proposed technique for a dialog system acting as a customer support service.Our results show the appropriateness of our technique to correctly detect and react to errors, enhancing the system performance and user satisfaction.

Keywords Error Detection and Correction · User Modeling · Spoken Dialog Systems · Human Machine Interfaces · Spoken Interaction.

1 Introduction

A spoken dialog system (SDS) is a software that accepts natural language as an input and produces natural language as an output engaging in a conversation

David Griol and José Manuel Molina
Group of Applied Artificial Intelligence (GIAA)
Computer Science Department
Carlos III University of Madrid (Spain)
Avda. de la Universidad, 30
28911 - Leganés (Spain)
E-mail: {david.griol, josemanuel.molina}@uc3m.es

with the user (McTear et al, 2016). To successfully manage the interaction with users, spoken dialog systems usually carry out five main tasks: automatic speech recognition (ASR), natural language understanding (NLU), dialog management (DM), natural language generation (NLG) and text-to-speech synthesis (TTS). These tasks are usually implemented in different modules. One of the main problems which must be considered during the interaction with a spoken conversational interface is the propagation of errors through the different modules in the system. These errors can cause the dialog system to misunderstand a user and select inappropriate system responses.

In this paper we propose a framework that can be incorporated in the architecture of a spoken dialog system to improve the error detection and correction tasks. Our proposal combines different components for user modeling, error detection, and generation of the system responses associated to different misunderstandings and non-understandings cases.

The proposed framework consists of three main components involved in the error handling process: the User-Behavior and User-Error models, the Error Detection Algorithm, and the Error Correction Mode. The main objective of the User-Behavior and User-Error models is to detect that an error has occurred or that the user wants to correct or modify previously provided information pieces. These statistical models can be trained with corpora of human-computer dialogs respectively related to the user responses that have made possible to achieve the dialog objectives or not for each specific dialog state. We propose to use a dialog simulation technique in order to automatically acquire the corpus that is required to learn the these two models.

The Error-Detection Algorithm considers the outputs provided by these two models to estimate the probability of the current user utterance including errors. The dialog manager takes this probability to select an error correction mode for which a specific strategy to inform the user and handle misunderstandings and non-understandings has been designed.

We have applied the proposed methodology to develop two versions of a practical dialog system acting as a customer support service to help solve simple and routine software/hardware repairing problems, both at the domestic and professional levels. The first dialog system uses the classical architecture of modules for a spoken dialog system previously described and the second one also integrates our proposed framework for error detection and correction. An in-depth comparative assessment of the developed dialog systems has been completed with recruited users. The results of the evaluation show that the use of our proposal allows to increase the number and quality of successful interactions and users' satisfaction with the dialog system.

The rest of the paper is organized as follows. In Section 2 we describe the motivation of our proposal and related work. Section 3 presents in detail our proposed framework for error detection and correction. Section 4 describes the practical application of our proposal. In Section 5 we discuss the evaluation results obtained by comparing two versions of the practical systems. Finally, in Section 6 we present the conclusions and outline guidelines for future work.

2 State of the art

In recent years, researchers have proposed the use of user modeling techniques to allow conversational interfaces to adapt their message according to the way they convey it to their interlocutors and the context in which the dialog takes place. The user simulation model consists of a user behavior model to generate synthetic user responses and an error simulation model, which simulates the speech recognition and understanding processes on the user intentions or utterances by the user behavior model. The following subsections summarize the main approaches proposed in the literature to generate these models.

2.1 User Behavior Modeling

Creating a user behavior model to interact with a spoken dialog system is a very challenging task, given the user simulations must be generative and attempt to replicate the variety of users interacting with the system (Williams, 2009). In addition, it is not obvious what data structures and input parameters are the most appropriate for modeling human behavior, which have to be usually learned with very little training data. A thorough literature review on the application of data mining techniques to user modeling and dialog system personalization can be found in (Schatzmann et al, 2006).

It is possible to classify the different approaches for user modeling with regard to the level of abstraction at which they model dialog. This can be at either the acoustic level, the word level or the intention-level. The latter is a particularly useful representation of human-computer interaction (Schatzmann et al, 2006). In recent years, simulation on the intention-level has been most popular, given that it is a particularly useful representation of human-computer interaction that avoids reproducing the enormous variety of human language on the level of speech signals or word sequences (Young et al, 2013).

Eckert et al (1997) introduced the use of statistical models to predict the next user action on the intentional level by means of a n-gram model. The proposed model has the advantage of being both statistical and task-independent. In (Levin et al, 2000), the bigram model is modified by considering only a set of possible user answers following a given system action. Both models have the drawback of considering that every user response depends only on the previous system turn. Therefore, the simulated user can change objectives continuously or repeat information previously provided. Scheffler and Young (1999) propose a graph-based model. The arcs of the network symbolize actions, and each node represents user decisions (*choice points*).

In (Schatzmann et al, 2007a), a technique for user simulation based on explicit representations of the user goal and the user agenda is presented. The user agenda is a structure that contains the pending user dialog acts that are needed to elicit the information specified in the goal. This model formalizes human-machine dialogs at a semantic level as a sequence of states and dialog acts. The agenda allows the user model to act with initiative. An

EM-based algorithm is used to estimate optimal parameter values iteratively. Schatzmann et al (2007c) uses the agenda-based simulator to train a statistical POMDP-based dialog manager.

More recently, Wang and Swegles (2013) proposed a technique that employs knowledge about the user's activity to disambiguate their spoken inputs. A Reinforcement Learning algorithm is used to acquire the knowledge and apply it for disambiguation. The interpreted user utterance is then transmitted to the dialog manager to select the next system response. Fonfara et al (2014) track hidden user goals using POMDPs, and Dethlefs and Cuayáhuitl (2015) have recently proposed a HMMs-based dialog simulation technique in which both user and system behaviors are simulated.

2.2 Error detection and correction

Most communication failures during the interaction with a spoken conversational interface are caused by ASR and SLU errors (Bulyko et al, 2005), unexpected conversational moves by the user (Aberdeen and Ferro, 2003), or complex and repetitive system prompts (Martinovsky and Traum, 2003).

Errors in communication generated during the ASR and NLU processes are associated to lengthy dialogs that can increase user frustration, generate more errors, and make more difficult error recovery if they are not detected and identified early (Bulyko et al, 2005). Confidence scores provided by the ASR and SLU modules are usually considered to detect possible errors and select system's confirmation turns (Kitaoka et al, 2003).

To avoid these errors, a basic solution is to improve the accuracy and robustness of the recognition and understanding processes (Wang et al, 2003; Erdogan et al, 2005; Gemello et al, 2006; Hakkani-Tur et al, 2006; López-Cózar et al, 2010). However, noisy environments and unexpected inputs make it impossible to develop perfect ASR and SLU modules. Thus, usually confidence scores generated by the ASR and SLU modules are used to optimize confirmation strategies. These scores, typically in the range [0.0, 1.0], can be computed at the phonetic, word and/or sentence levels, and help determine whether the recognized words and corresponding dialog acts are correct. However, confidence scores are not entirely reliable and are dependent on user types and noisy environments. In addition, when the confidence score exceeds the threshold but an error actually occur, it is not usually easy for the user to correct the decisions taken by the dialog manager and put the dialog back on track.

For these reasons, robust dialog management is a challenging topic. Error handling approaches during the dialog management process involve different mechanisms to detect that the user wants to correct or change an entry, to change the dialog manager to an error correction mode, and to generate spoken responses that are appropriate to that situations (Lee et al, 2010).

The dialog manager usually adopts two types of confirmation strategy for error recovery: explicit confirmation and implicit confirmation (Skantze, 2009).

These confirmation strategies are useful to repair unreliable information using confidence scores on the described levels. With explicit confirmation, the system generates an additional dialog turn to explicitly confirm the data item(s) obtained from the previous user turn. The main disadvantage of explicit confirmations is that they can make the interaction less efficient and even excessively repetitive if all the data items provided by the user have to be confirmed.

Implicit confirmations include part of the user's previous input in its next question. If the user answers the question directly, then it is assumed that the given information is correct. However, this kind of confirmations can lead to the user producing utterances that are beyond the scope of the ASR and SLU components.

Non-understanding situations, which occur when the system has not been able to collect any data from its interaction with the user, are usually handled by the dialog manager asking the user to repeat the input, or asking for it to be rephrased (Shin et al, 2002). The traditional rephrase strategy consists of asking the user just to repeat the previous utterance. However, this cannot always correct the recognition and understanding errors (e.g., out-of-vocabulary and out-of-grammar problems). This strategy can be improved by providing help messages to speak well-recognizable and well-understandable utterances adapted to the current dialog state (Fukubayashi et al, 2006). Karsenty and Botherel (2005) proposed adaptable and adaptive transparency strategies to help users respond appropriately to system errors; these strategies were applied to a system that provided plane and train schedules.

In addition, several studies have also verified that it is useful to exploit N-best recognition and understanding hypotheses rather than a single hypothesis to determine the current dialog state (Kitaoka et al, 2003). The dialog manager can then consider the different options and the associated confidence scores to reestimate the dialog state for the selection of the next system action.

Different studies have also been focused on the relationships between the user behavior and different characteristics of the system response (Bulyko et al, 2005). The main conclusions obtained are that system rephrasing is a useful strategy in error spirals, encouraging user rephrasing tends to lead to lower WER, and system apologies reduce user frustration in error conditions.

3 Our Proposal for error detection and correction in Spoken Dialog Systems

Figure 1 shows the architecture that integrates our proposed framework for error detection and correction in spoken dialog systems. As can be observed, the proposed framework consists of three main components involved in the error handling process: the User-Behavior and User-Error models, the Error Detection Algorithm, and the Error Correction Mode.

The main objective of the User-Behavior and User-Error models is to detect that an error has occurred or that the user wants to correct or modify previously provided information pieces. These statistical models can be trained

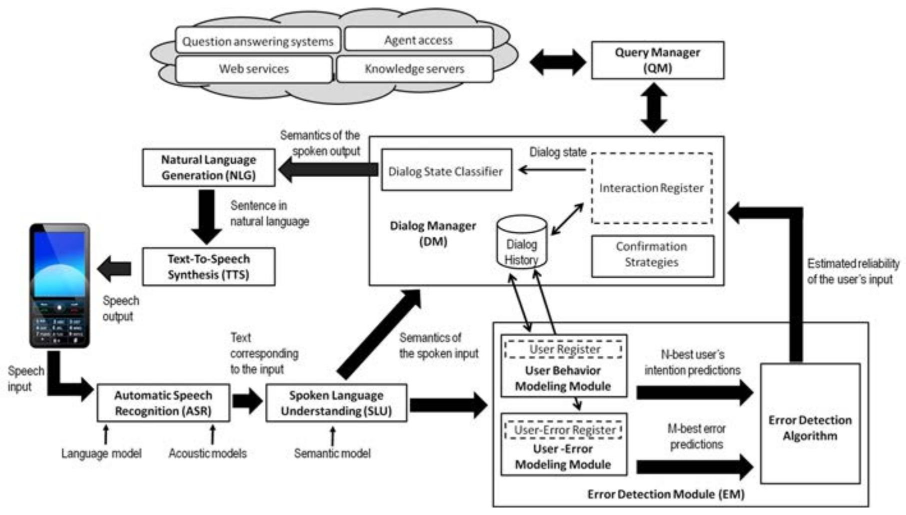
Question answering systems
Agent access
Web services
Knowledge servers
Query Manager (QM)
Semantics of the spoken output
Natural Language Generation (NLG)
Dialog State Classifier
Dialog state
Interaction Register
Sentence in natural language
Dialog Manager (DM)
Dialog History
Confirmation Strategies
Text-To-Speech Synthesis (TTS)
Estimated reliability of the user's input
Speech output
Text corresponding to the input
Semantics of the spoken input
User Register
User Behavior Modeling Module
N-best user's intention predictions
Speech input
Automatic Speech Recognition (ASR)
Spoken Language Understanding (SLU)
Error Detection Algorithm
User-Error Register
User -Error Modeling Module
M-best error predictions
Language model
Acoustic models
Semantic model
Error Detection Module (EM)

Fig. 1 Architecture of a spoken dialog system with the proposed framework for error detection and correction

with corpora of human-computer dialogs. The goal of the User-Behavior model is to provide user responses that have been annotated in the training corpora as valid ones for the current dialog situation (i.e., the provision of these user responses for the specific dialog state have made possible to achieve the required objectives at the end of the dialog). The User-Error model is trained with the user responses for each specific dialog situation that have generated the selection of inappropriate system responses by the dialog manager (e.g., misunderstoods, dialog failures, reprompts, provide help, etc.).

The Error-Detection Algorithm considers the outputs provided by these two models to estimate the probability of the current user utterance including errors. The dialog manager takes this probability to select an error correction mode for which a specific strategy to inform the user and handle misunderstandings has been designed. The following subsections detail the main components of the proposed framework.

3.1 The User-Behavior and User-Error Models

The statistical methodology that we propose to learn the user-behavior and user-error models employs a data-driven classification procedure to generate abstract representations of the user turns taking into account the previous history of the dialog. Our approach has the advantage of taking into account the data supplied by the user model throughout the dialog, which makes the estimation of the statistical model tractable, without causing scalability problems. It is also portable across application domains and interaction languages and can be incrementally optimized to tackle complex dialog tasks.

Another salient characteristic is the inclusion of a data structure that stores the information provided by the user model. The main objective of this structure is to easily encode the complete information related to the task provided by the user during the dialog history, then considering the specific semantics of the task and including this information in the proposed classification process.

The user models simulate the user intention level, that is, they provide concepts and attributes that represent the intention of the user utterance. Therefore, these models carry out the functions of the ASR and NLU modules of a spoken dialog system. The user responses are generated taking into account the information provided by the corresponding model throughout the history of the dialog, the last system turn, and the objective(s) predefined for the dialog.

In order to control the interaction, our user models use the representation the dialogs as a sequence of pairs (A_i, U_i), where A_i is the output of the dialog manager (the system response) at time i, expressed in terms of dialog acts; and U_i is the semantic representation of the user turn (the result of the understanding process of the user input) at time i. This way, each dialog is represented by $(A_1, U_1), \cdots, (A_i, U_i), \cdots, (A_n, U_n)$, where A_1 is the greeting turn of the system (the first turn of the dialog), and U_n is the last user turn. We refer to a pair (A_i, U_i) as S_i, the state of the dialog sequence at time i.

The lexical, syntactic and semantic information associated to the speaker u's ith turn (U_i) is usually represented by:

- the words uttered;
- part of speech tags, also called word classes or lexical categories. Common linguistic categories include noun, adjective, and verb, among others;
- predicate-argument structures, used by SLU modules in various contexts to represent relations within a sentence structure. They are usually represented as triples (subject-verb-object).
- named entities: sequences of words that refer to a unique identifier. This identifier may be a proper name (e.g., organization, person or location names), a time identifier (e.g., dates, time expressions or durations), or quantities and numerical expressions (e.g., monetary values, percentages or phone numbers).

In this framework, we consider that, at time i, the objective of the user model is to find an appropriate user answer U_i. This selection is a local process for each time i and takes into account the sequence of dialog states that precede time i, the system answer at time i, and the objective of the dialog $\mathcal{O}$. If the most probable user answer U_i is selected at each time i, the selection is made using the maximization:

$$\hat{U}_i = \operatorname*{argmax}_{U_i \in \mathcal{U}} P(U_i | S_1, \cdots, S_{i-1}, A_i, \mathcal{O})$$

where set $\mathcal{U}$ contains all the possible user answers.

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 dialog preceding time i).This data structure, that we call *User Register* (UR), contains the information provided by the user throughout the previous history of the dialog. After applying the above considerations and establishing the equivalence relations in the histories of the dialogs, the selection of the best U_i is given by:

$$\hat{U}_i = \underset{U_i \in \mathcal{U}}{\operatorname{argmax}} P(U_i | UR_{i-1}, A_i, \mathcal{O})$$

As previously described, the User-Behavior and User-Error statistical models are learned dividing the training corpus in the user responses annotated as appropriated or not for the specific dialog situations (i.e., they are trained with different corpus). As in our work on dialog management (Griol et al, 2014), we propose the use of a multilayer perceptron (MLP) to make the assignation of a user turn. The values of the output layer can be viewed as the a posteriori probability of selecting the different user answers defined for the simulator given the current situation of the dialog. The choice of the most probable user answer of this probability distribution leads to the previous equation. In this case, the user simulator will always generate the same answer for the same situation of the dialog. Since we want to provide the user simulator with a richer variability of behaviors, we base our choice on the probability distribution supplied by the MLP on all the feasible user answers.

The main objective of the Error-Detection Algorithm at each time i is to compare the outputs provided by the User-Behavior and User-Error models with the output provided by the SLU module for the current user utterance (U_i). Considering that the User-Behavior model provides the *N-best* user responses for the current dialog state and the Error-behavior model provides the *M-worst* user responses, the position of the output provided by the SLU module in these lists is considered for the Expected and Error functions:

$$\mathit{Expected}\ (U_i) = 1 - (\frac{1 - pos}{N})$$

$$\mathit{Error}\ (U_i) = 1 - (\frac{1 - pos}{M})$$

The Error-Detection Algorithm provides a number in the range $[-1, 1]$ according to the following expression:

$$\mathit{Expected\ vs.\ Error}\ (U_i) = \mathit{Expected}(U_i) - \mathit{Error}(U_i)$$

where the values nearest to -1 indicate the possibility of the current user utterance containing errors and the values nearest to 1 estimate the non-presence of errors.

3.1.1 Training the user models

We propose the use a dialog simulation technique in order to automatically acquire the corpus that is required to learn the user behavior and user error models (Griol et al, 2013). This approach is based on the interaction of a user simulator and a dialog manager simulator.. Both modules initially use a random selection of one of the possible answers defined for the semantics of the task (user and system dialog acts). At the beginning of the simulation, the responses are defined as equiprobable. When a successful dialog is simulated, the probabilities of the responses selected by the dialog manager simulator and the user simulator during that dialog are incremented before starting a new simulation. This way, the described statistical dialog model is gradually learned with the incorporation of the successfully simulated dialogs. Although this approach converges to collaborative dialogs in the long term, it does not relay in predefined rules or user profiles, which would produce a more restricted behavior.

An error simulation module is implemented to include semantic errors in the generation of dialogs. Once the user model has selected the information to be provided to the dialog manager simulator, the error simulator modifies the frames created and provides a confidence score for each concept and attribute in the semantic representation of the user turn (Schatzmann et al, 2007b).

Both processes are carried out separately following the noisy communication channel metaphor by means of a generative probabilistic model $P(c, U_a|\tilde{U_a})$, where U_a is the true incoming user dialog act, $\tilde{U_a}$ is the recognized hypothesis, and c is the confidence score associated with this hypothesis.

On the one hand, the probability $P(\tilde{U}_a|U_a)$ is obtained by Maximum-Likelihood using the initial labeled corpus acquired with real users. To compute it, we consider the recognized sequence of words w_U and the actual sequence uttered by the user $\tilde{w}_U$. This probability is decomposed into a component that generates the word-level utterance that corresponds to a given user dialog act, a model that simulates ASR confusions (learned from the reference transcriptions and the ASR outputs), and a component that models the semantic decoding process.

$$P(\tilde{U}_a|U_a) = \sum_{\tilde{w}_U} P(U_a|\tilde{w}_U) \sum_{w_U} P(\tilde{w}_U|w_U) P(w_U|U_a)$$

Confidence score generation is carried out by approximating $P(c|\tilde{a}_u, a_u)$ assuming that there are two distributions for c. These two distributions are handcrafted, generating confidence scores for correct and incorrect hypotheses by sampling from the distributions found in the training data corresponding to our initial corpus.

$$P(c|U_a, \tilde{U}_a) = \begin{cases} P_{corr}(c) & if \quad \tilde{U}_a = U_a \\ P_{incorr}(c) & if \quad \tilde{U}_a \neq U_a \end{cases}$$

3.2 Managing the interaction

The statistical methodology that we propose to develop domain-independent dialog managers is described in (Griol et al, 2014). The methodology is based on the estimation of a statistical model from the sequences of system and user dialog acts obtained from a set of training data. The selection of the following system answer is based on a classification process that takes into account the history of the dialog. The codification of this information allows the system to automatically generate a specialized answer that takes the current situation of the dialog into account.

Task-dependent information is isolated from the model so the methodology can be applied to develop dialog managers for any application domain. This is done by defining an interaction register (IR) that takes into account whether the user has provided a given piece of information related to the task and also the confidence scores that the recognition and understanding modules have assigned to this piece of information. This allows not only to cope with the situations observed the training corpus, but also to manage unseen situations by selecting the most convenient system action.

We improve this methodology by also considering the output of the proposed framework for error detection as an additional input for the dialog manager. Table 1 shows the set of task-independent error handling strategies designed to generate specialized system responses. As it can be observed, the different cases have been classified into system responses for handling misunderstandings and system responses for handling non-understandings.

4 Practical Application

Our proposal has been applied to evaluate a customer support dialog systems that helps solving simple and routine software/hardware repairing problems, both at the domestic and professional levels.

The definition of the system's functionalities and dialog strategy was carried out by means of the analysis of 250 human-human conversations (7215 user turns) provided by real assistants attending the calls of users with a software/hardware problem at the City Council of Leganés (Madrid, Spain). The labeling defined for this corpus contains different types of information, that have been annotated using a multilevel approach similar to the one proposed in the Luna Project (Stepanov et al, 2014). The first levels include segmentation of the corpus in dialog turns, transcription of the speech signal, and syntactic preprocessing with POS-tagging (i.e., word-category disambiguation) and shallow parsing (i.e., identification of the different constituents in a sentence). The next level consists of the annotation of main information using attribute-value pairs. The other levels of the annotation show contextual aspects of the semantic interpretation.

The attribute-value annotation uses a predefined domain ontology to specify concepts and their relations. The attributes defined for the task include *Concept*, *Computer-Hardware*, *Action*, *Person-Name*, *Location*, *Code*, *TelephoneNumber*, *Problem*, etc.

Dialog act (DA) annotation was performed manually by one annotator on speech transcriptions previously segmented into turns. The DAs defined to label the corpus are the following: i) Core DAs: *Action-request*, *Yes-answer*,

Error Handling Strategy	**Examples**
Strategies for handling misunderstandings	
Explicit confirmation	Did you say you that the model is Officejet Pro 8610?
Implicit confirmation	So the red light is flashing ...
Strategies for handling non-understandings	
Notify that a non-understanding happened	Sorry, I did not catch that ...
Ask user to repeat	Could you please repeat that?
Ask user to rephrase	Could you please rephrase that?
Repeat prompt	Have you checked your Internet connection?
Give a you-can-say help message	For instance, you could say something like "The printer is not working" or "I have a problem with my modem"
Give an explain-more help message	Right now I need you to tell me what is your technical difficulty.
Give a full help message	Right now I need you to tell me what is your technical difficulty. For instance, you could say something like "The printer is not working" or "I have a problem with my modem"
Give tips about how to best interact with the system	Okay, I know this conversation is not going well. There are things you can try to help me understand you better. Speak clearly and naturally: do not speak too quickly or too slowly. Give short, concise answers. Calling from a quiet place helps. If you would like to start from scratch, you can say 'start-over' at any time.
Ask for a short answer and repeat prompt	Please use shorter answers because I have trouble understanding long sentences... Have you checked your Internet connection?
Ask for a short answer and give a you-can-say a help message	Please use shorter answers because I have trouble understanding long sentences... For instance, you could say something like "The printer is not working" or "I have a problem with my modem"
Ask user to speak less loud and repeat prompt	I understand people best when they speak softer. Have you checked your Internet connection?
Fall the current request and move-on	Sorry, I did not catch that. I assume your Internet connection is working, now we can check...
Yield the turn	[...] (system remains silent, yielding the turn to the user)
Ask user if they would like to start over	I am sorry I am still having trouble understanding you, and I might do better if we restarted. Would you like to start over?
Give up	I am sorry but I am having lots of trouble to understand you and I do not think I will be able to help you. One moment please while I transfer you to a human operator.

Table 1 Task-independent error handling strategies defined for the dialog management module

No-answer, *Answer*, *Offer*, *ReportOnAction*, *Inform*; ii) Conventional DAs: *Greet*, *Quit*, *Apology*, *Thank*; iii) Feedback-Turn management DAs: *ClarificationRequest*, *Ack*, *Filler*; iv) Non interpretable DAs: *Other*.

The original FrameNet[1] description of frames and frame elements was adopted for the predicate-argument structure annotation, introducing new frames and roles related to hardware/software only in case of gaps in the FrameNet ontology. Some of the frames included in this representation are *Telling*, *Greeting*, *Contacting*, *Statement*, *Recording*, *Communication*, *Being operational*, *Change operational state*, etc.

[1] https://framenet.icsi.berkeley.edu/fndrupal/

The basic structure of the dialogs is usually composed by the sequence of the following tasks: *Opening*, *Problem-statement*, *User-identification*, *Problem-clarification*, *Problem-resolution*, and *Closing*. This set of tasks contains a list of subtasks, such as *Problem-description*, *Problem-Request*, *Help-Request*, *Problem-Confirmation*, *Brand-Identification*, *Model-Identification*, *Name-Identification*, *Message-Confirmation*, *Resolution-Confirmation*, etc. The shared plan is represented as a data register that encapsulates the task structure, dialog act structure, attribute-values and predicate-argument structure of utterances. Figure 2 shows the transcription of a dialog included in the training corpus.

A_1: Welcome! How may I help you?
U_1: Hello, my printer does not print correctly.
A_2: Tell me your name and surname.
U_2: My name is Ricardo Santamaria.
A_3: Can you tell me your phone extension?
U_3: Nine four sixty-three.
A_4: Can you tell me the brand and model?
U_4: HP Officejet Pro 8610.
A_5: Is the printer on now?
U_5: It is on.
A_6: Can you tell me the message on the printer display?
U_6: Paper jam is written.
A_7: Can you see the active printer from the control panel of your PC?
U_7: It is active.
A_8: Try the following steps. Say next after each step. Turn the printer off, and then on.
U_8: Next.
A_9: Print a Printer Status Report.
U_9: Next.
A_{10}: Has this process solved your problem?
U_{10}: No.
A_{11}: Remove any loose paper from the main and optional trays.
U_{11}: Next.
A_{12}: Clear any jammed paper from the rear of the printer.
U_{12}: Next.
A_{13}: Has this process solved your problem?
U_{13}: Yes.
A_{14}: Anything else?
U_{14}: No, thank you.
A_{15}: Most of these steps will also work for troubleshooting an office copier. Have a nice day!

Fig. 2 Example of a dialog acquired with recruited users and the practical dialog system (translation from Spanish to English)

During the *Problem-statement* task, the caller explains the problem the reasons why he/she calls the help-desk. In the *User-identification* task, the operator asks for additional information regarding the identity of the caller. Once the caller has described the problem, the operator can ask for additional information to clarify it during the *Problem-clarification* task.

During the *Problem-resolution* task, the operator asks the user to perform specific tests. We have defined nine different subtasks inside this generic segment, given that our goal is to detect not only that the dialog is in this segment, but also what are the specific problem that has to be resolved: *Printer* (P4), *Network connection* (P5), *PC going slow* (P6), *Monitor* (P7), *Keyboard* (P8),

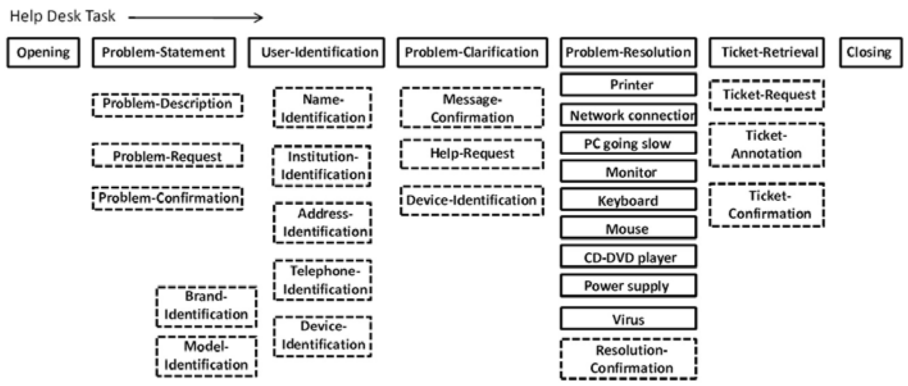

Fig. 3 Incremental evolution of the dialog structure

Mouse (P9), *CD-DVD player* (P10), *Power supply* (P11), and *Virus* (P12). The operator assigns a ticket number for the current call if the problem has not been solved after this task. The user must take note of this number and inform about this to the operator. The dialog ends at the *Closing* phase, in which the operator also tries to give a useful advice related to the described problem. Figure 3 shows an example of the incremental evolution of dialog structure with the complete set of tasks and substasks. It can be observed the difficulty of correctly detecting the complete structure of the dialog.

The complete set of human-human dialogs were manually labeled including this task/subtask information. This information was incorporated for each user and system turn in the dialogs. Two versions of the dialog system have been developed. The first one (*Generic System*) uses the structure of dialogs described in Figure 1 without the integration of our proposed framework for error detection and correction. The second one (*Enhanced System*) uses the sames modules of the *Baseline System*, but also integrates the proposed framework.

5 Evaluation of the proposed methodology

We have completed a comparative evaluation of the two practical dialog systems developed for the task using a set of 25 scenarios covering the different problems that users can formulate to the system. A total of 300 dialogs were recorded from interactions of 20 recruited users for our department employing the two dialog systems. These users selected the specific scenarios in a random order and using a random assignment of both systems. An objective and subjective evaluation were carried out.

The following measures were defined in the objective evaluation to compare the dialogs acquired with the dialog systems (Schatzmann et al, 2005; Ai et al, 2007):

– High-level dialog features: These features evaluate how long the dialogs last, how much information is transmitted in individual turns, and how active the dialog participants are. We have considered the average number of turns per dialog, the percentage of different dialogs without considering the attribute values, the number of repetitions of the most seen dialog, the number of turns of the most seen dialog, the number of turns of the shortest dialog, and the number of turns of the longest dialog.
– Dialog style/cooperativeness measures: These measures try to analyze the frequency of different speech acts and study what proportion of actions is goal-directed, what part is taken up by dialog formalities, etc.
– Task success/efficiency measures: These measures study the goal achievement rates and goal completion times.

To compare the different versions of the system we computed the mean value for the previously described evaluation measures, which we extracted from different studies (Ai et al, 2007; Schatzmann et al, 2005). We then used two-tailed t-tests to compare the means across the different types of scenarios and users as described in (Ai et al, 2007). The significance of the results was computed using the SPSS software with a significance level of 95%.

5.1 High-level Dialog Features

Table 2 presents the results of this evaluation. As can be observed, the different systems could interact correctly with the users in most cases, achieving success rates higher than 88% in a difficult domain in which only spoken interaction is provided. However, the real context-aware system obtained a higher success rate, improving the baseline system results by 6% absolute. This difference showed a significance value of 0.025 in the two-tailed *t*-test.

The confirmation error correction rates were also improved in absolute values by using the *Enhanced System*. Both results can be explained by the fact that the proposed framework provides a specific strategy to improve the recognition or understanding processes and decrease the error rate. From the results obtained for these measures, it can also be observed the importance of considering the dialog context for error detection and correction. This relationship was also significant in the *t*-test (significance value of 0.027). A problem occurred in several cases in which the user input was misrecognized but it had high confidence score, in which case it was forwarded to the dialog manager by the error detection and correction method. However, as the success rate shows, this problem did not have a remarkable impact on the performance of the dialog systems.

It can also be observed that when *Enhanced System* was used, there was a reduction in the average number of turns and in the number of turns in the longest, shortest and most observed dialogs. These results show that the use of the proposed framework for error correction and detection made it possible to reduce the number of necessary system actions to attain the dialog goals for the different tasks. Although the reduction in the average number of dialog

	Generic System	*Enhanced System*
Dialog success rate	88.0%	94.0%
Average number of turns per dialog	25.1	23.2
Percentage of different dialogs	84.6%	88.7%
Repetitions of the most observed dialog	4	3
Average number of actions per turn	1.2	1.5
Number of user turns of the most observed dialog	12	10
Number of user turns of the shortest dialog	8	6
Number of user turns of the longest dialog	13	11
Confirmation rate	38%	29%
Error correction rate	0.78%	0.94%

Table 2 Results of the high-level dialog measures for the two systems

turns between the real context-aware system and the baseline system was significant (significance value of 0.016), the comparative between the results of the longest, shortest and most seen dialogs for the three systems provided a non-significant value in the *t*-test. In addition, the results show a higher variability in the dialogs generated with the *Enhanced System* as there was a higher percentage of different dialogs and the most observed dialog was less repeated. There was also a slight increment in the mean values of the turn length for the dialogs collected with the *Enhanced System* due to the better selection of the system actions in the improved strategy.

The dialogs acquired with the *Generic System* have a higher standard deviation (4.34) given that the proportion of number of turns per dialog is more disperse. The dialogs gathered with the *Enhanced System* have a smaller deviation (3.91) since the successful dialogs are usually those which require the minimum number of turns to achieve the objective(s) predefined in the scenarios.

Regarding the dialog participant activity, the dialogs acquired with the *Enhanced System* have a higher proportion of system actions, as less confirmations are required using this system. There is also a slight reduction in the mean values of the turn length, these dialogs are statistically shorter, as they provide 1.36 actions per user turn instead of the 1.48 actions provided by the *Generic System.* This is again because the users have to explicitly provide and confirm more information in the *Generic System.* The results of the *t*-test in a comparative analysis of this measure showed a significant difference of 0.030.

5.2 Dialog style and cooperativeness

Dialog style and cooperativeness measures show the frequency of different speech acts and reflect the proportion of actions that are goal-directed (i.e. not indexed in dialog formalities). Figures 4 show the frequency of the most predominant user and system dialog acts in the dialogs acquired with the three systems. On the system side, we have measured the confirmation of concepts and attributes, questions to require information, and system answers generated after a database query. We have not take into account the opening and closing system turns. On the user side, we have measured the percentage of turns in which the user carries out a request to the system, provide information, confirms a concept or attribute, Yes/No answers, and other answers not included in the previous categories (e.g, dialog formalities or out of task information).

In both cases, it can be observed that there are significant differences in the distribution of dialog acts. Figure 4 shows that users need to provide less information using the *Enhanced System*. This explains the higher proportion for the rest of user actions with regard the *Generic System* (both differences significant over 98%). There is also a higher proportion of yes/no actions for the dialogs acquired with the *Enhanced System*. These actions are mainly used to confirm that the specific queries have been correctly provided using context information.

Figure 4 also shows that there is a reduction in the system requests when the *Enhanced System* is used. This explains a higher proportion of the inform and confirmation system actions when this system is used in comparison with the *Generic System* (significant difference over 98%).

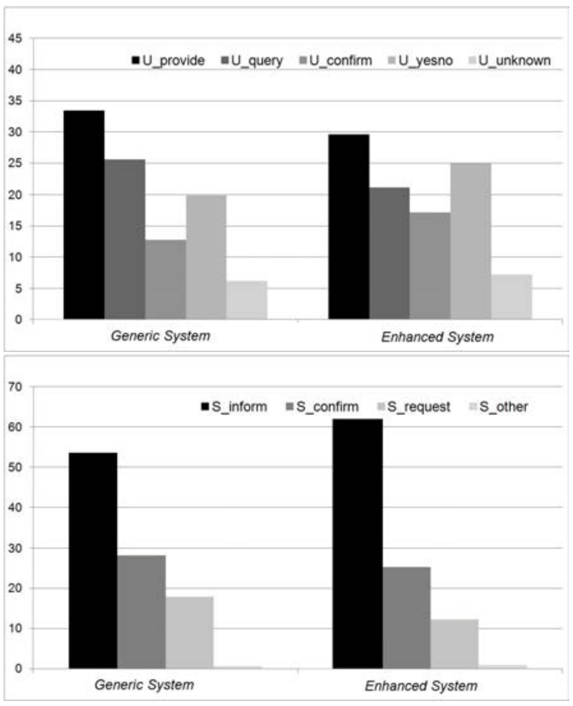

Fig. 4 Histogram of user dialog acts (top) and system dialog acts (bottom)

Finally, we have measured the proportion of goal-directed actions (request and provide information) versus the grounding actions (confirmations) and rest of actions. We grouped all user and system actions into three categories: "goal directed" (actions to provide or request information), "grounding" (confirmations and negations), and "other". Table 3 shows a comparison between these categories. As can be observed, the dialogs provided by the *Enhanced System* have a better quality, as the proportion of goal-directed actions is higher than the values obtained for the *Generic System*.

	Generic System	*Enhanced System*
Goal-directed actions	67.11%	73.21%
Grounding actions	32.02%	25.88%
Rest of actions	0.87%	0.91%

Table 3 Proportions of dialog spent on-goal directed, ground and other actions

5.3 Subjective assessment

We also asked the users to complete a questionnaire to assess their subjective opinion about the system performance. The questionnaire had eight 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: *If the system made errors, was it easy for you to correct them?*; vi) Q6: *How much did you feel that the system cares about you?*; vii) Q7: *How much did you trust the system?*; viii) Q8: *With which frequency would you continue working with the system?* The possible answers for each one of the questions were the same: Never/Not at all, Seldom/In some measure, Sometimes/Acceptably, Usually/Well, and Always/Very Well. All the answers were assigned a numeric value between one and five (in the same order as they appear in the questionnaire). The following subsections present the results obtained for the four types of evaluation metrics previously described.

Table 4 shows the average results of the subjective evaluation using the described questionnaire. It can be observed that the users perceived that the *Enhanced System* understood them better. Moreover, they expressed a similar opinion regarding the easiness for correcting system errors. They also mentioned that it was easier to obtain the information specified for the different objectives using the *Enhanced System*, and that the interaction with the system was more adequate with this system. Ratings of satisfaction, ease of use, trust, and desire to continue using the system were also improved by the *Enhanced System*. Together, these results indicate that the proposed framework represents a viable and promising medium for error handling in spoken dialog systems.

	Generic System	*Enhanced System*
Q1	M = 4.12, SD = 0.77	M = 4.24, SD = 0.42
Q2	M = 3.95, SD = 0.64	M = 4.32, SD = 0.46
Q3	M = 3.44, SD = 0.56	M = 4.11, SD = 0.37
Q4	M = 3.13, SD = 0.45	M = 3.94, SD = 0.41
Q5	M = 3.37, SD = 0.58	M = 4.43, SD = 0.31
Q6	M = 3.73, SD = 0.71	M = 3.84, SD = 0.31
Q7	M = 4.03, SD = 1.02	M = 4.22, SD = 0.56
Q8	M = 3.81, SD = 0.43	M = 3.96, SD = 0.35

Table 4 Results of the subjective evaluation (For the mean value M: 1=worst, 5=best evaluation)

6 Conclusions

Dialog systems are currently very widespread and are employed for an increasingly higher number of applications. This is to great extent due to the improvements achieved in their constituent modules (e.g. higher speech recognition accuracy, more fine-grained speech understanding, etc.). However, they are not extent of errors, and error transmission through the different operation phases provokes system malfunctions that have a negative impact on user satisfaction.

In this paper, we present a framework for the detection and correction of errors in spoken dialog systems, which can be integrated in the classic architecture of these systems regardless of their application domain. The framework incorporates two statistical models trained over a dialog corpus. The first model takes into account the valid user responses for each dialog state, that is, the responses that allow a correct development of the dialog where users achieve their objectives. The second model considers the unsatisfactory/incorrect user responses for each dialog state.

An algorithm has been developed to estimate the presence of errors in the current user utterance by combining the best hypotheses generated by both models. This estimation is considered as an additional input of the dialog manager and used to compute whether it is necessary to correct possible errors. We have designed an strategy differentiating between the main misunderstanding and non-understanding scenarios, so that the dialog manager can provide an acceptable tailored response when entering the error correction state.

Our proposal has been evaluated with a dialog system corresponding to a technical help-desk domain, for which we have implemented a state-of-the-art dialog manager and a dialog manager that incorporates our proposal. The results show that the use of our framework improves the dialogs generated with more accurate error detection and the selection of a higher number of goal-directed system responses, which have a positive impact on user satisfaction. As future work, we would like to apply our proposal to application domains in which system errors may have a very negative impact on users, such as tutoring systems, and evaluate it with different speech recognizers of disparate performance.

Acknowledgements This work was supported in part by Projects MINECO TEC2012-37832-C02-01, CICYT TEC2011-28626-C02-02, CAM CONTEXTS (S2009/TIC-1485).

References

Aberdeen J, Ferro L (2003) Dialogue patterns and misunderstandings. In: Proc. ISCA Workshop on Error Handling in SDSs, pp 17–23

Ai H, Raux A, Bohus D, Eskenazi M, Litman D (2007) Comparing Spoken Dialog Corpora Collected with Recruited Subjects versus Real Users. In: Proc. SIGdial, pp 124–131

Bulyko I, Kirchhoff K, Ostendorf M, Goldberg J (2005) Error-correction detection and response generation in a spoken dialogue system. Speech Communication 45(3):271–288

Dethlefs N, Cuayáhuitl H (2015) Hierarchical reinforcement learning for situated natural language generation. Natural Language Engineering 21(3):391–435

Eckert W, Levin E, Pieraccini R (1997) User modeling for spoken dialogue system evaluation. In: Proc. of ASRU, pp 80–87

Erdogan H, Sarikaya R, Chen S, Gao Y, Picheny M (2005) Using semantic analysis to improve speech recognition performance. Computer Speech and Language 19:321–343

Fonfara J, Hellbacha S, Bohme H (2014) Imitating dialog strategies under uncertainty. In: Proc. of IHCI, pp 131–138

Fukubayashi Y, Komatani K, Ogata T, Okuno H (2006) Dynamic help generation by estimating user's mental model in spoken dialogue systems. In: Proc. International Conference on Spoken Language Processing, pp 1946–1949

Gemello R, Mana F, Albesano D, Mori RD (2006) Multiple resolution analysis for robust automatic speech recognition. Computer Speech and Language 20:2–21

Griol D, Carbo J, Molina JM (2013) An automatic dialog simulation technique to develop and evaluate interactive conversational agents. Applied Artificial Intelligence 27(9):759–780

Griol D, Callejas Z, López-Cózar R, Riccardi G (2014) A domain-independent statistical methodology for dialog management in spoken dialog systems. Computer, Speech and Language 28(3):743–768

Hakkani-Tur D, Bechet F, Riccardi G, Tur G (2006) Beyond ASR 1-best: using word confusion networks in spoken language understanding. Computer Speech and Language 20(4):495–514

Karsenty L, Botherel V (2005) Transparency strategies to help users handle system errors. Speech Communication 45:305–324

Kitaoka N, Kakutani N, Nakagawa S (2003) Detection and recognition of correction utterance in spontaneously spoken dialog. In: Proc. Eurospeech, pp 625–628

Lee C, Jung S, Kim K, Lee GG (2010) Hybrid approach to robust dialog management using agenda and dialog examples. Computer Speech and Language 24(4):609–631

Levin E, Pieraccini R, Eckert W (2000) A stochastic model of human-machine interaction for learning dialog strategies. IEEE Transactions on Speech and Audio Processing 8(1):11–23

López-Cózar R, Callejas Z, Griol D (2010) Using knowledge of misunderstandings to increase the robustness of spoken dialogue systems. Knowledge-Based Systems 23

Martinovsky B, Traum D (2003) The error is the clue: breakdown in human-machine interaction. In: Proc. ISCA Workshop on Error Handling in SDSs, pp 11–17

McTear MF, Callejas Z, Griol D (2016) The Conversational Interface. Springer

Schatzmann J, Georgila K, Young S (2005) Quantitative Evaluation of User Simulation Techniques for Spoken Dialogue Systems. In: Proc. SIGdial, pp 45–54

Schatzmann J, Weilhammer K, Stuttle M, Young S (2006) A Survey of Statistical User Simulation Techniques for Reinforcement-Learning of Dialogue Management Strategies. Knowledge Engineering Review 21(2):97–126

Schatzmann J, Thomson B, Weilhammer K, Ye H, Young S (2007a) Agenda-Based User Simulation for Bootstrapping a POMDP Dialogue System. In: Proc. of HLT/NAACL, pp 149–152

Schatzmann J, Thomson B, Young S (2007b) Error simulation for training statistical dialogue systems. In: Proc. of ASRU, Kyoto, Japan, pp 526–531

Schatzmann J, Thomson B, Young S (2007c) Statistical User Simulation with a Hidden Agenda. In: Proc. of SIGdial, pp 273–282

Scheffler K, Young S (1999) Simulation of human-machine dialogues. Tech. rep., CUED/F-INFENG/TR 355, Cambridge University Engineering Dept.

Shin J, Narayanan S, Gerber L, Kazemzadeh A, Byrd D (2002) Analysis of user behavior under error conditions in spoken dialogs. In: Proc. of ICSLP, pp 2069–2072

Skantze G (2009) Exploring human error recovery strategies: implications for spoken dialogue systems. Speech Communication 45(3):325–341

Stepanov E, Riccardi G, Bayer A (2014) The Development of the Multilingual LUNA Corpus for Spoken Language System Porting. In: Proc. of LREC, pp 2675–2678

Wang F, Swegles K (2013) Modeling user behavior online for disambiguating user input in a spoken dialogue system. Speech Communication 55:84–98

Wang Y, Acero A, Chelba C (2003) Is word error rate a good indicator for spoken language understanding accuracy? In: Proc. of ASRU, pp 577–582

Williams J (2009) The best of both worlds: Unifying conventional dialog systems and pomdps. In: Proc. of Interspeech, pp 1173–1176

Young SJ, Gasic M, Thomson B, Williams JD (2013) Pomdp-based statistical spoken dialog systems: A review. Proc of IEEE 101(5):1160–1179