Strategies to cope with errors in human-machine spoken interactions: using chatbots as back-off mechanism for task-oriented dialogues

Chapter · January 2016

CITATIONS
5

READS
731

2 authors:

Andreea Ioana Niculescu
Institute for Infocomm Research
43 PUBLICATIONS **304** CITATIONS
SEE PROFILE

Rafael E. Banchs
Agency for Science, Technology and Research (A*STAR)
163 PUBLICATIONS **2,080** CITATIONS
SEE PROFILE

Strategies to cope with errors in human-machine spoken interactions: using chatbots as back-off mechanism for task-oriented dialogues

Andreea I. Niculescu, Rafael E. Banchs

Institute for Infocomm Research, (I²R), A*STAR, Singapore

{andreea-n,rembanchs}@i2r.a-star.edu.sg

Abstract

In this paper we present an overview on different types of errors that can occur while interacting with speech devices. First, we propose a series of strategies on how to deal with these errors and make recommendations on how to use speech technology in more efficient ways. Then, we focus our discussion on a specific strategy for dealing with errors by using chatbots as back-off mechanism for task-oriented dialogues.

Index Terms: natural language processing, spoken dialogue systems, task-oriented dialogue, chatbots, errors, human computer interaction, interaction design

1. Introduction

Speech is the most natural form of communication between people: it is fast, transparent, easy to use, and requires little practice. Movies about the future often depict computers using speech to interact with humans creating a false image of powerful, intelligent machines with human traits. In reality, speech remains the holy grail of the human-computer interaction technology being a challenge due to prevalent errors and inability of both parties to cope with them.

In this paper we analyse a set of errors occurring during the interaction with speech interfaces and offer solutions on how to deal with these errors.

We describe the concept of error in the context of task-oriented dialogue interaction. Here, the goal achievement can be seen as a successive transactional chain of verbal statements that leads the users to achieve certain objectives. For each objective there is an optimal path of successive transactions. As such, an error can be described as a deviation from the optimal solution path [1].

The rest of the paper is structured as follows. First, in section 2, we describe the main type of errors commonly occurring in task-oriented dialogue interactions, including both machine and human errors. Then, in section 3, we focus our attention on main strategies to deal with errors. Two cases are considered: preventive strategies (i.e. dealing with errors before they happen) and corrective strategies (i.e. dealing with errors after they happen). Then, in section 4, we describe our proposed strategy of using chatbots as back-off mechanism for task-oriented dialogues, and present specific examples of it in the context of a restaurant recommendation and booking system. Finally, in section 5, we present our conclusions and plans for future work in this area.

2. Errors in speech interactions

In this section, we describe the main type of errors commonly occurring in task-oriented dialogue systems. In our approach we assume errors to be already detected by the system and as such, we will focus our attention on those errors made by machines, followed by a discussion on the type of errors made by humans.

1. Errors made by machines

Using speech in interactive tasks remains still difficult, merely because of the high complexity of the background information a dialogue system has to deal with: user inputs are typically required to be put in context by using world knowledge and common sense. Unfortunately, this cannot be modelled reliably with the current state-of-the-art technology as it imposes some serious challenges in terms of computational power and external resources needed for the processing.

Another problem often encountered is related to the accuracy of linguistic unit segmentation, i.e. properly detecting the boundaries of phones, syllables, words and sentences. For example, “I scream” and “ice cream” sounds very similar not only to machines, but also to human listeners.

The variability in acoustic channels, due to different microphones, rooms acoustics, background noise and timber across speakers, is an additional source of errors in speech processing. Also, the linguistic ambiguity when dealing with homophones (“two” vs “too”) or semantical expressions “crispy rice cereal” vs. “crispy rice serial” constitute significant sources of error in dialogue [2].

Although most of these acoustic and lexical errors are naturally handled and corrected by humans by means of semantic and pragmatic knowledge, machines cannot cope with these errors efficiently. This is, as mentioned above, mainly because of the lack of proper mechanisms to model world knowledge and common sense.

2. Errors made by humans

Even thought using speech to communicate with others is natural and does not require too much effort, people have different styles of speaking. This is, in particular, relevant when talking to machines because, depending on their background, experience and expectations, people tend to adopt different interaction styles. For example, people with technical background and low expectations towards speech technology tend to use keywords, while people with non-technical background and no concrete expectations tend to use sentences predominantly. Errors will happen when there is a mismatch between users' speaking style and the way the system is designed to work: keywords vs. sentences [3].

Also, talking too slow, or too fast, can elicit truncated messages or unit segmentation problems that further lead to speech processing errors.

Overestimating the system capabilities and asking out-of-domain questions that challenge the system boundaries is also a source or errors in the communication process [4]. As it will be discussed later in section 4, our proposed strategy of using a chatbot engine as a back-off mechanism in a task-oriented dialogue, is especially relevant to address this particular problem.

Similarly, asking in-domain questions that were not designed to be used during a particular interaction exchange, as well as using synonyms or different word ordering, constitutes significant sources of errors too.

Additionally, failing to issue a command during the time allocated for asking a question is also considered a human error, more specifically, a "void-input" error [1].

3. Strategies for dealing with errors

Eliminating errors completely from speech interactions is an impossible task. Even for humans, errors in speech interaction are prevalent. Therefore, understanding errors per se could help increasing the naturalness and realism of the interactions with a dialogue system, provided they are handled properly: while we cannot completely avoid errors, we can design the human-computer interaction in such way that errors can be either minimised or handled graciously once they happened.

In this section, we focus our attention in two types of strategies: preventive and corrective strategies. In the first case, we describe strategies used for dealing with errors before they happen, while in the second case, we describe strategies for dealing with errors after they happen. Later, in section 4, we will describe in detail our proposed corrective strategy, which is based on using a chatbot to disguise task-oriented dialogue errors.

1. Dealing with errors before they occur

A strategy for preventing errors before they occur is to complement speech with other channels of communication by means of multimodality. Such multimodal channels include lips reading, gesture recognition, haptic and graphic interfaces, among others. The data fusion allowed by multimodal inputs minimizes the risk of errors, ensuring the correct message is transmitted to the system. Different modalities complement each other to compensate shortcomings [5].

Another strategy for preventing errors from happening is context disambiguation. It is often the case that entity names, such as geographical areas or organisations are misrecognised or not recognised at all. Therefore, in case of doubt the system should offer users with the option to input the referred name by typing. By disambiguating entities and locations early in the transactional chain, the system avoids further errors.

It is also important to properly signal or inform the differences in the interaction style if they differ from what the user might expect from previous interactions. Further differences in expectation towards the system caused by possible inferences by analogy from related task domains may invite users to ask clarifying out-of domain questions that the system cannot handle.

Example 1

System: *"Okay, I have completed your booking: five concert tickets for Saturday, May 16, at 7:00PM"*

System: "*Do you need any additional assistance*"

User: "*Would it be possible to get a discount?"*

In this case, the user wants a discounted rate for the concert tickets, but she does not know that such an option is not available after the booking has been completed. Thus, the system should take into account the user's expectations by mentioning during the booking process what are the discount options available.

Another important consideration refers to the information load; this should be wisely distributed across the interaction flow. Especially, the first system prompt to the user has an important role for the entire interaction: it should contain as much information as possible about what the system provides and how the user should interact with the system [6].

Example 2

System: "*Hello, welcome to R3: Restaurant Reservation and Recommendation System. You can search for restaurants in Singapore according to the following criteria: type of cuisine, price range, city area and opening hours*.

Also, if the dialogue flowchart has a complicated structure which requires several user inputs, or the user himself has changed his input several times, the system should briefly repeat the commitments made earlier in order to keep the user informed on the current status of the task under consideration [7].

Ensuring a proper way of expression constitutes another strategy to deal with errors before they occur. In general, too open or non-specific formulations should be al-

ways avoided. Such formulations, apart from inviting the user to take the initiative and ask out-of-domain questions, may lead users to hesitation, false starts and/ or revisions. Karsenty suggested that using explicit requests helps users to better structure their responses and to avoid long utterances [8].

Additionally, system's prompts should also be short, as far as possible. However, in some particular cases, system's prompts cannot be short. In such cases, prompts should contain a dialogue focus at the end 'pointing' to the next dialogue sequence [7].

If the system needs time to process the information, it should inform the user that it will take a few seconds to provide the requested information [6]. This strategy also helps avoiding overlapping speech, which is highly problematic for speech recognition. In cases when overlapping speech cannot be prevented, a recommend-ed strategy is to avoid simultaneous talk by immediately turning the system silent if interrupted by the user. Commonly, detection of silence sequences is used to establish the end of a turn. In these cases, overlapping occurs when long pauses from the user are mistakenly interpreted as turn-endings by the system [9].

Last but not least, highlighting differences that exist between the interlocutors is another strategy that helps minimizing error occurrences in spoken dialogue. Such differences between interlocutors are likely to influence the interaction course significantly. When learning to speak, people implicitly learn what to expect from a 'standard' conversational partner but, on the other hand, when interacting with a 'non-standard' interlocutor, people adjust their manner of speaking according to their perceived partner's abilities. Some examples of these situations include cases such as when speaking to small children, talking to impaired people, or even between 'standard' interlocutors who find themselves in very noisy environments [3]. In a similar way, the computer is in many respects a 'non-standard' partner. Then, it is recommended to explicitly highlight this partner asymmetry to avoid miscommunication. This can be achieved by providing clear indications about the system competence. Research has demonstrated that people who tend to make more well-formed phrases use a reduced vocabulary when they assume they are talking with a machine [10, 11].

2. Dealing with errors after they occur

Once an error has happened, a meta-communication strategy needs to be initialized in order to bring the verbal transaction to the correct path. Generally, the sooner an error is detected the better. A good practice to detect errors at early stages is to provide immediate feedback by incorporating key information from user input in the answer.

Example 3

User: *"I need a table for two at the Little Italian Trattoria, please."*

System: "*Sure, there are plenty of good Italian restaurants I can recommend, what area of the city are you interested in?*"

User: *"No, I want to make a reservation at the Little Italian Trattoria."*

Very often, strategies requesting for clarifications tend to be difficult to handle, as they may raise unplanned questions. In such cases, the system needs to handle the situation by presenting alternative input options. If no input comes from the user, it is advisable to repeat the inquiry options not more than twice, and to break off the conversation by saying goodbye and redirecting the user to a human information source. It is desirable that the system does not hang up abruptly [6].

Also helpful in situations when errors occur is to keep a history of verbal transaction accessible for users. This enables both the user and the system to be aware of the context and consequently allowing for detecting and reverting to the specific point of miscommunication.

Finally, another strategy for dealing with errors after they occur is using humor. Although humour does not improve the quality of the dialogue in terms of objective goal completion, its use can be beneficial as it allows the system to gain time for recovery purposes while providing the user with an enhanced user experience. Rather than making evident system errors or excessively confirming unresolved user's intentions, a witty answer helps to relax the stress in the communication process. The user will probably understand the system misunderstood the question and repeat the input. However, it is likely that the user will be more tolerant to failures if the system is perceived as behaving more human-like than expected.

4. Dealing with errors with chatbots

In this section we describe our proposed strategy for dealing with errors after they occur by means of integrating a chatbot with a task-oriented spoken dialogue system.

First, we present the fundamental concepts behind the proposed mechanism, followed by specific examples within the context of a dialogues system for restaurant recommendation and reservation.

1. Proposed strategy details

Unlike task-oriented dialogues which focus on human-system collaboration to accomplish specific predefined goals, the chat-oriented dialogue is not aiming to achieve any specific objective: its purpose is to provide pragmatically and semantically meaningful responses to keep the interaction going. By design, task-oriented dialogue engines are implemented to provide deep natural language understanding capabilities into a very restricted topic or domain. On the other hand, chatbots

are designed to cover a wide variety of topics and domains while having very shallow understanding capabilities. In general, coverage and understanding constitute trade-off capabilities in spoken dialogue system design.

As already discussed in a previous section, an important source of errors in task-oriented dialogue systems is the production of out-of-domain utterances. While this is a common and natural practice in human-human dialogue, it constitutes a serious problem for human-machine dialogue. On one hand, the inability to deal with out-of-domain input represents a strong limitation of current dialogue technologies from the usability point of view.; on the other hand, designing a task-oriented dialogue system able to handle every possible out-of-domain input is at the moment an impossible task.

Therefore, we propose to complement the specificity of task-oriented dialogue systems with the more general , broad coverage of chatbots. The main objective of this strategy is to create the illusion of a the dialogue engine that can handle out-of-domain inputs. It is important to understand that this strategy does not improve the performance of the task-oriented dialogue in terms of goal completion; however, it is likely to improve the user experience.

In terms of implementation the chat-oriented system can be combined as a slave system with the main task-oriented system either in a cascade or a parallel architecture. In the first case, the chat engine is used to produce a response only when the main system is unable to handle the user input. In the second case, the chat engine would always produce a response; here an additional decision module is used to decide when and how to produce a chatbot response.

2. Proposed strategy example

In this section, we show how our strategy can be used to improve the user experience of a conventional task-oriented dialogue system. The strategy helps dealing with out-of-domain user inputs and reduces the necessity of system confirmations and verifications.

The examples are extracted from dialogue exchanges between a human user and a task-oriented dialogue engine in the context of restaurant recommendation and reservation [12]. The dialogue system is augmented with an example-based chat-oriented dialogue system trained with dialogues from movies [13]. The engines are working in a parallel systems architecture that allows system outputs to be used in combination in an attempt to make system verifications and repetitions more human-like.

The following example shows a typical interaction with the task-oriented dialogue system without the intervention of the chat-oriented dialogue engine:

Example 4

User: *"Please suggest a good Italian restaurant."*

System: "*What area of the city do you want me to search for?*"

User: *"Near Sentosa"*

System: *"Okay, I have found one restaurant for you: Alkaff Mansion Ristorante. Do you want more information or do you want to make a reservation?"*

User: *"Yes Please"*

System: *"Okay, I will do a reservation for you! And, how many guests is this reservation for?"*

User: *"Only two persons."*

System: *"And, will you be interested in dinner or lunch?"*

User: *"Please make it for tonight."*

System: *"Your reservation has been made"*

In this case the recommendation and reservation process is completed successfully without any problem, basically because the user behaved as the system "expected".

Next example illustrates two cases of user "misbehavior" that put the system into trouble; here the the chatbot engine intervenes to back-off the main dialogue engine in an attempt to solve an out-of domain occurrence and make the system be perceived as more capable.

Example 5

System: *"Okay, I have found one restaurant for you: Alkaff Mansion Ristorante. Do you want more information or do you want to make a reservation?"*

User: *"Do you like French food more than Italian?"*

System: *"Do you want me to search for French food instead?"*

The task-oriented system decides to consider the out-of-domain utterance as a change in preference. At this point the user would need to answer "no" or restate her original intention on having Italian food in order to continue with the reservation step.

In the next example, the system is able to provide an answer with high confidence. Here the chatbot is used in combination with the task-oriented engine to improve the quality of the response and to bring back the conversation topic to the domain of interest.

Example 6

System: *"Okay, I have found one restaurant for you: Alkaff Mansion Ristorante. Do you want more information or do you want to make a reservation?"*

User: *"Do you like French food more than Italian?"*

System: *"Anything but guts and mushrooms. Do you want more information about Alkaff Mansion Ristorante or do you want to make a reservation?"*

In the following example we show how the chatbot response is used to completely replace the answer of the task-oriented dialogue system. Here the error source is not necessarily an out-of-domain input, but merely related to the system incapacity of dealing with complex

answers. Again, we show first the outcome of the task-oriented system, without the intervention of the chatbot.

Example 7

System: *"Okay, I will do a reservation for you! And, how many guests is this reservation for?"*

User: *"I will go with my grandma."*

System: *"Sorry, could you please let me know how many guests is this reservation for?"*

User: *"We will be two persons."*

System: *"And, will you be interested in dinner or lunch?"*

User: *"Please make it for tonight."*

System: *"Your reservation has been made"*

Obviously, the system is not able to extract the information related to the number of guests from the provided user input. The system needs to restate the question and the error become evident. Let us consider now the outcome of the chatbot intervention.

Example 8

System: *"Okay, I will do a reservation for you! And, how many guests is this reservation for?"*

User: *"I will go with my grandma."*

System: *"You are lucky you still have a grandma, mine passed away five years ago."*

User: *"Oh, I am sorry to hear that."*

System: *"And, will you be interested in dinner or lunch?"*

User: *"Please make it for tonight."*

System: *"And, how many guests is this reservation for?"*

User: *"Two people, my grandma and me."*

System: *"Your reservation has been made"*

As seen from this example, the chatbot is able to provide a meaningful response to the user input while covering the incapability of the task-oriented dialogue engine to properly resolve the user input. Only when the system asks again for the same information the user realises the system did not understand the initial request. The user still needs to restate the number of guests, but the overall interaction experience seems to be improved.

Even though chatbots appear to be an attractive solution for enhancing error handling strategies and overall system usability further user studies are needed to conform our suppositions and validate this approach. Additionally, the chatbots (as any other dialogue system) can also be prone to errors, such as incomplete, inappropriate or unmatching answers that can lead to even more complex dialogue exchanges. Thus, a further strategy to deal with such situations has to be elaborated in the future.

5. Conclusions and future work

In this paper we presented an overview on different types of errors that can occur in human-computer interaction by means of speech and dialogue. First, we showed different types of errors sources and discussed various recovery strategies. Further, we focused a specific strategy of dealing with errors, namely using chatbots. In our proposed strategy we use chatbots as back-off mechanism for task-oriented dialogues allowing for a much more natural flow of interaction and potentially improving the overall user experience despite inherent failures. As such, the strategy combines the specificity of task-oriented dialogue systems and the generality and coverage broadness of chatbots. We illustrated the proposed strategy by using a parallel implementation of a task-oriented dialogue system in the restaurant recommendation and reservation domain and a chat-oriented dialogue engine trained with dialogue from movies.

Although the proposed strategy looks promising, there are some few challenges to overcome in order to achieve a robust integration of the two systems. These challenges include the normalization of confidence scores across both task- and chat-oriented dialogue engines, the jointly use of dialogue state information from both engines and by both engines, and the capability of automatically tuning the parameters of the decision mechanisms that generates the final system response.

Another important aspect is the evaluation; while our current observations suggest that the strategy can help to improve the overall user experience with the system, we do not have reliable methods to measure variations in performance with respect to the variations in the implementation decisions and parameters. Developing reliable evaluation protocols for chat-oriented dialogue and user satisfaction will definitively be an interesting field for research in the years to come.

Finally, it is also very important to integrate use centric design principles from the beginning. Successful strategies for error prevention and handling in human-computer interaction systems should take into account the many other aspects discussed in section 3, including the user's background knowledge and expectations, the proper distribution of the information load, highlighting partner asymmetry, avoiding speech overlap and ensuring an appropriate expression manner, as well as providing feedback, in addition to humor.

6. References

[1] A. Oulasvirta, K.P. Engelbrecht, A. Jameson and S. Möller. 2006. The Relationship between User Errors and Perceived Usability of a Spoken Dialog System. SCA/DEGA Tutorial and Research Workshop on Perceptual Quality of Systems.

[2] C. Munteanu and G. Penn. 2012. Speech-based interacttion. Course, ACM SIGCHI

[3] A.I. Niculescu. 2011. Conversational interfaces for task-oriented spoken dialogues: design aspects influencing interac-

tion quality. PhD thesis, University of Twente. CTIT No. 11-212 ISBN 978-90-752-9600-6.

[4] A.I. Niculescu, M.Q. Lim, S.A. Wibowo, K.H Yeo, B.P. Lim, M. Popow, D. Chia and R.E. Banchs. 2015. IDA - An Intelligent Driver Assistant for Smart Parking in Singapore. Proc. of INTERACT 2015.

[5] T. Falck, S. Gamm, and A. Kerner. 1993. Multimodal dialogues make feature phones easier to use. In Proc. Applications of Speech Technology, pages 125–128, Budapest, 1993.

[6] M. Cohen, J. Giangola, and J. Balogh. 2004. Voice User Interface Design. Addison Wesley, New York.

[7] N.O. Bernsen, H. Dybkjaer, and L. Dybkjaer. 1998. Designing interactive speech systems. From first ideas to user testing. Springer Verlag, NY.

[8] L. Karsenty. 2002. Shifting the design philosophy of spoken natural language dialogue: From invisible to transparent systems. International Journal of Speech Technology, 5(2):147–157.

[9] A. Hjalmarsson. 2009. Human interaction as a model for spoken dialogue system behaviour. PhD thesis, KTH Royal Institute of Technology, Stockholm, Sweden.

[10] N. Fraser and N. Gilbert. 1991. Effects of system voice quality on user utterances in speech dialogue systems. In Proc. of EUROSPEECH, pages 57–60.

[11] A. Hauptmann and A. Rudnicky. 1998. Talking to computers: An empirical investigation. International Journal of Man-Machine Studies, 28(6):583–604.

[12] S. Kim and R.E. Banchs. 2014. "R-cube: a dialogue agent for Restaurant Recommendation and Reservation", in Proc. of Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)

[13] R.E. Banchs and H. Li. 2012. IRIS: a chat-oriented dialogue system based on the vector space model, in Demo Session of ACL, pp. 37–42.