The Need For Robustness In Dialog Systems

Susanne Ekeklint and Fredrik Kronlid

Term paper for Speech Technology 1, autumn 2001
GSLT - Graduate School of Language Technology

1 Introduction

Dialog systems are moving out of the research labs and out into the real world. This change is important for the performance of the systems. The users in the real world don't (in general) know how the system works and are not even necessarily familiar with computers. The non-human environment becomes less controllable - variable sound conditions, different types of noise etc. This means that dialog systems, in order to be usable, must be robust enough to handle the new users and the new environments. The more flexibility the system has, the harder the requirements get on almost every component.

2 What is a dialog system?

A dialog system is, as its name suggests, a system that can engage in a dialog. Glass (1999) says that dialog systems are "systems [that] allow users to interact with a machine to retrieve information, conduct transactions, or perform other problem-solving tasks". This is a very broad definition of the term dialog system, since this would classify almost any program with any kind of user interface as a dialog system.

What we will refer to as dialog systems in this paper, are spoken dialog systems. In a spoken dialog system the user has the choice of interacting with the computer by using his/her voice. The computer may also interact by using voice. The main purpose of a spoken dialog system is to provide an interface between a user and a computer-based application such as a database or expert system (McTear, 2001). Examples of expert system are systems for timetable information, weather forecasts (McTear 2001) and translation services (Wahlster, 2000).

A dialog system can use several different channels for interaction. There is an increasing interest for multi modality in dialog systems. Some of the channels for multi modality dialogs are keys, voice, visualisations and audio. In many cases the support from the visual modality in speech reading can make a difference between failure and success in communications, according to Granström (1999).

3 What is robustness?

Definition of the word robust given in the Oxford English Dictionary:

> "Robust [...] Applied to a statistical test that yields approximately correct results despite the falsity of certain of the assumptions on which it is based; also, to a calculation, process, or result if the result is largely independent of certain aspects of the input."

The result, mentioned in the dictionary entry, could also be seen as the desired outcome of an interaction with a dialog system. Given the results in (Lippmann, 1997) it seems like the construction of an ASR (Automatic Speech Recogniser) that can give an exact decoding of each and every possible utterance is still a far-away goal - and therefore the Dialog Manager and the Natural Language Understanding unit must be robust enough to be able to interpret the user's contribution even if it is to some extent distorted.

We believe that this means that a robust system should be able to continue a "conversation" even if it doesn't understand some of the information given by the user. It shall always aim to understand what the user means. For example, if some of the words aren't recognised, the system should either ask for more information or be able to retrieve the information by analysing the surrounding context of the word. Another thing that shouldn't stop the system from functioning is the use of some (from the system's point of view) unknown grammatical construction. A spoken dialog system that isn't speaker independent should also be able to perform independent of the speakers' dialect, age or sex.

The features mentioned above are valid as robustness parameters for any speech recognition application - not only for dialog systems.

Another, more NLP-connected, definition of what robustness is:

> ”A kind of monotonic behaviour, which should be guaranteed whenever a system is exposed to some sort of non-standard input data: A comparatively small deviation from a predefined ideal should lead to no or only minor disturbances in the system's response, whereas a total failure might only be accepted for sufficiently distorted input” (Mentzel, 1995).

Of course, the dialog manager and the natural language understanding unit can make mistakes – even humans misunderstand utterances from time to time. One good thing about humans is that they often realize that something has gone wrong when they have misunderstood or misinterpreted something. In our opinion this qualification is one of the most desired qualities in dialog systems. Humans also have the ability to guide themselves and their discussion partners out of the dead-end that was caused by the misunderstanding/-interpretation. The robustness of a dialog system is heavily dependent on its ability to detect dead-ends and recover from them. We will return to this issue in chapter 4.2.

Another robustness feature, not covered by the definition above, is the system's ability to understand, and prevent, unrealistic expectations of the system. This includes both overestimating the capacity of the system as well as posing questions outside the system's domain.

4 What makes a dialog system robust?

To be able to decide if a system is robust enough there are several aspects to take into consideration. One has to consider the performance of the overall system, as well as the performance of individual components that it consists of. There is also a need for tools to evaluate different aspects of robustness. Here we will discuss these areas, starting with a discussion about the users, since a poor model of the user's intentions can be a threat to any system (Lüth 1994). After that discussion there will be a presentation of individual components that a dialog system consists of and how these can be made more robust.

4.1 The users

Dialog systems are often used as replacement for some kind of information service, formerly conducted by humans. When humans are expected to retrieve information by talking to a computer instead of a human it is important that they still receive good service.

When comparing the participants in a dialog human-to-human with the participants engaged in a human-computer dialog there are several differences. It seems clear that people are independent entities, capable of independently motivated action, and it also seems that they often take each other's actions, motivations and other mental attitudes into consideration when acting, particularly for tasks such as dialogs (Allwood et al. 2000). If a conversation is to be working well, it is important that all the participants have the same goal with the conversation. If a user starts asking questions that isn't part of the domain of the dialog system it is important that the dialog is put back on right track again. The system has to take initiative so that the conversation again aims for the goal. It is also important that the user understand what the system is capable of discussing.

Faced to a spoken dialog system, naive users do not always behave as expected. They might not aim for a specific goal and they may not have the intentions that the system developers assumed. This means that the system will be asked questions outside its domain. We will demonstrate this by using the August system as an example:

The August system is a spoken dialog system; created partly with the goal to collect spontaneous speech input from people with little or no previous experience of spoken dialog systems (Bell and Gustafson, 1999). August was installed at public locations and people were invited to interact with the system. Results from the August experiments show that a large part of the communication with the system has more social than information-seeking content. There are also examples of users trying to test the system's general knowledge about the world by asking questions like "What is two plus two?" (Bell and Gustafson, 1999). This is clearly outside the domains of most dialog systems (including August's).

Bell and Gustafson (1999) says that most social interactions are relatively simple form a linguistic point of view, and that such a domain could easily be added in order to make the system more robust. There isn't so easy on the other hand to foresee exactly what knowledge of the world that users would like to test.

The dialog system evaluation tool PARADISE (Walker et al. 1998) uses some robustness parameters to calculate an estimation of the user satisfaction. This is important - a robust system (that is usable in other ways) gives satisfied users (See chapter 6 for further discussion about PARADISE).

4.2 The robustness of individual components

We will now discuss different components in a generalised spoken dialog system. The components are discussed in terms of problems and improvements. There are many ways in how the different components can be implemented; this chapter is only to point out some of the mechanisms in some components that we find important in terms of robustness.

There are ways of measuring the performance of the individual components. Evaluating speech recognition and language understanding components are relatively easy, but it is more difficult when it comes to the dialog manager. The standard way to evaluate speech recognition and language understanding is to, given a certain input, compare the output to the desired output. Since the dialog manager has a wider range of valid output, this kind of evaluation is harder to do (McTear 2001). McTear also points out, that it is possible to achieve perfect sentence understanding even with non-perfect sentence recognition.

Speech recognition

One of the main areas that need further development in dialog systems is speech recognition. Many researchers believe that recognisers will enjoy widespread use and become commonplace only if their performance approaches that of humans under everyday listening environment (Lippmann, 1997).

Two studies concerning speech recognition that have been presented by Lippmann (1997) demonstrates that error rates of humans are much lower than those of machines in quiet, and that error rates of current recognisers increase substantially at noise levels which have little effect on human listeners. Results also shows that adaptation algorithms can improve the recognising performance when noise spectra are known during training.

Another way to increase robustness is to minimize the length of the user's contributions. Speech recognition works more reliably on short utterances, in other words - shorter contributions leads to lower error-rate. This can be obtained by changing the dialog manager into a more restrictive model (McTear, 2001). Of course the change has implications on the overall usability of the system (see below under Dialog management). Also, shorter contributions are less complex, and lead to lower demands on the language understanding parts of the system.

One further thing that clearly could improve the robustness of the speech recognition is more knowledge on how to handle accents, dialects and different manners of speaking in different circumstances (Aust and Lenke, 1997).

Language understanding

Many systems use HPSG (Head-driven Phrase Structure Grammar) parsers (and alike) in order to parse the input. HPSG parsers are capable of giving very good and detailed analyses of input text, but are not robust enough to be relied on as the single method to analyse input text.

If an HPSG parser doesn't succeed in analysing the input text, it doesn't give partial solutions, but simply says, as the response to the failed Prolog query, 'no'. Another robustness threatening feature of HPSG parsers is that if the parse is successful, a vast number of analyses are returned (Glass 1999). As a backup solution, if the HPSG parser fails, a stochastic parser, that gives fewer (1) and less fine-grained analyses, is used. This has as the consequence that the natural language unit must work with an analysis that doesn't contain all the information that it could have done - of course a potential source of errors.

This can be augmented to the model that Verbmobil uses - a set of stochastic parsers that give one analysis each, and a voting for the best analysis is conducted (Whalster, 2000).

Language generation

When an utterance has been understood by the system it is time for the system to respond on it. If a system is a multi-modal one, which has a graphic interface, it might be satisfactory to simply display a table of information. If a system only responds by speech synthesis it is more difficult. Sometimes the information that matched the request consists of a long table of data. In these cases the system has to summarize or select significant data and present it to the user. Glass (1999) draws the conclusion that; language generation used by most spoken dialog systems tends to be static, using a constant response pattern. Further, he says that he has observed that introducing variation in the way to prompt users with additional queries (e.g., "Is there anything else you'd like to know?", "Can I help you with anything else?", "What else?") is quite effective in making the system appear less robotic and more natural to users.

Components in the language generator typically identifies text of different kind, such as digits, acronyms and names. The input can normally be mixed with different kinds of information, such as phonetic text or special symbols that controls either system functions or the linguistic/phonetic processing (Carlsson and Granström, 1997). Using digits the wrong way can really confuse the user, for example if a date in it is spelled out like an amount.

The linguistic processing module is one part of the language generator. It varies a lot in complexity and level of ambition when comparing systems. One of the entities that the processing module consists of is the syntactic analyser. Its complexity can vary from key-word identifications to complete sentence parsing (Carlsson and Granström, 1997).

Speech synthesis

The type of speech synthesis used in dialog systems varies widely, from pre-recorded phrases, to word and phrase concatenation, to general-purpose synthesizers (Glass, 1999). Several of the systems today have been developed on basis of phonological theories. They are build as special rule compilers that operate on rule-by rule or segment-by-segment.

The quality of speech synthesis varies a lot in different systems. The quality and the intelligibility of speech are usually very difficult to measure. It is however important to be able to evaluate speech synthesis in several aspects. If one likes to compare different competing systems it would be useful to have some kind of standard to measure the systems by. It is also important from a developer's point of view to be able to obtain the system's weaknesses for further developing of the system (Carlsson and Granström, 1997).

Glass (1999) says that the speech synthesis component is the one that often leaves the most lasting impression on users – especially when it is not especially natural.

Dialog manager

The dialog managers' main task is to make the dialog run smoothly. As mentioned earlier, the system must detect and handle dead-ends, or alternatively make it easy for the user to detect them and signal this fact to the system. Hence, it is important for the system to communicate to the user its beliefs of the user's intentions and goals. The system must be able to switch the dialog flow in case of recognition problems.

The system's ability to detect problems is of course one of the most important things to consider, when deciding whether or not a system is robust. An example of this serious problem for dialog systems is that if one user contribution has been rejected (the system didn't believe that it understood what the user wanted), the next utterance is also likely to be misunderstood (Glass 1999). This puts the user in a bad position, since it is highly likely that the subsequent utterance will be misinterpreted as well.

In Walker et. al. (1998) there is a comparison of how a dialog system behaves when confronted with novice users compared to novice users who had gone through a tutorial. It is striking that the system performs much better in all aspects when confronted with a person who knows the system. It is not desirable that all potential users takes a tutorial before using the system – one reason for creating dialog systems is that they should offer a user interface which is natural and easy to use. Thus the dialog system must communicate what it can and can't do. Preferably it should understand when it's asked to do something that it's not capable of doing.

Many systems have some kind of confidence scoring of the recognized utterances (Glass 1999). When the confidence score is low enough (the system believes that it has misunderstood the user's input) it goes into some kind of emergency mode. If a system that allows a rather free dialog runs into problems, it may back off to a more directed dialog in order to pass by the problematic passage (Glass 1999). Another strategy is to let a low confidence scoring trigger some kind of help message.

Code modularisation of the different turns that the dialog may take, reduces the amount of code that the system needs. This is a better way to handle large dialogs since it speeds up the system and also increases the flexibility of the dialog (Aust and Lenke, 1997). Obviously, a flexible dialog is more error-prone then one that is restricted and that gives the user menu-like choices.

Sub-dialogs can be used to handle dialogs that is outside the system's original domain. Sub dialogs can be used to guide users around difficult parts of the dialog. A sub dialog can also be activated when a specific word is being used. For example if the user says *Help* it will start a sub dialog that guides the user in some way, usually it gives some kind of explanation of how the system works.

Graphical user interface

It is not unusual that there is a graphic interface to complement the speech synthesis. For example if there are several different choices or a lot of information that the user has to take into consideration, it is helpful to complement the system with graphics. Another purpose is that the interface continuously can feed back information that the system has obtained from parsing the users utterances.

Waxholm is a system that provides information about boat traffic in Stockholm archipelago. The Waxholm system uses their graphic interface to show for example time given by the user and the data from the database that matches the given time (Carlsson and Granström, 1995). When showing the time given by the user the system shows that it has understood what the user said. By visualising the timetable that matched the users wishes it is easier for the user to get an overview, especially if there are a lot of information. To give the user an overview of what the system can provide, the graphic interface is build like a micro world that represents the system's domain in terms of pictures. This way the system gives the user a realistic picture of what it can handle. It also gives the user a frame for what he or she might expect.

Speed

Speed is of course not a separate component in a dialog system but it is important to take into consideration when building components. To have a good dialog the user should not have to wait for the system to respond. If the system holds on an answer there is place for misunderstandings. The user might think that the system didn't understand and might repeat the question. We also believe that if the system is too slow at responding it gives the impression of being unintelligent. Our believe concerning speed are backed up by Aust and Lenke, they mention speed several times in an article about the Philip dialog system. They mean that the system is not good enough if it is not able to give answers at run-time. That demand we think can be a bit harsh, it is however not to much to ask for some kind of response or reflective feed-back on an utterance at run-time.

5 Verification strategies for dialog systems

Two verification strategies are described by (McTear, 2001), implicit and explicit verification. The two strategies can be illustrated with the responses to the user utterance: "I want to go from Trento to Milano".

The explicit strategy means posing questions for verification after each user input action, such as the question "do you want to go from Trento to Milano". The implicit verification means embedding facts (or what the system has understood as a fact) in the following question, such as "at what time do you want to go from Trento to Milano".

The explicit verification strategy is more robust than the implicit one, but leads to a more artificial (and tiresome) dialog. According to (McTear, 2001) the implicit strategy is preferable when the system is "reasonably confident with the output from the speech recognition and language understanding components". This is due to the fact that this kind of implicit verification requests that are elicited from the system gives rise to a wider range of possible responses that may be too much to handle for the recognition and the language understanding components.

When a dialog system doesn't understand what the user says and fails to match the input to a meaningful utterance, the standard response seems to be something like "I did not understand, please rephrase". This is a message or request that can be tiresome, and it isn't very informative either.

When a dialog system is telling the user that it does not understand the user's contribution, the user often tries to spell out the word, or to emphasize the syllables in the word – usually with a poor result (Glass 1999). Such an interaction shows 5 robustness deficits of the system:

1. The system failed to recognize the word.
2. The system didn't realize what it didn't understand.
3. The system failed to inform the user what word it didn't recognize
4. The system failed to inform the user about how things should be said in order for it to understand the user's input.
5. The system didn't understand the 'helpful' contribution of the user.

6 Robustness evaluation

One of the greater parts of the dialog system is the dialog manager. The total score for the dialog systems robustness can be measured in terms of the dialog manager's ability to perform both implicit and explicit recovery when the speech recognition or the language-understanding unit fails. A measure for the implicit recovery (IR), which is the dialog module's capacity to regain utterances which are partially failed at recognition or understanding levels, can be obtained by dividing the number of times the dialog manager manages to calculate the correct analysis of the utterance by the number of partially unrecognised utterances (Danieli and Gerbino, 1995).

The metric ER is mentioned in the paper by Danieli and Gerbino (1995), but is not described. It is claimed that this is a metric of the dialog system's robustness, but we consider this being a metric of the dialog manager's robustness, since the dialog system consists of more units than the dialog manager. One shouldn't judge a dialog system that has almost no need for IR/ER as inferior to a system that has excellent values for ER/IR, but needs it a lot.

Another model, or tool, for the evaluation of spoken dialog systems is called PARADISE. It claims to describe the overall user satisfaction. It breaks down the term user satisfaction into costs and success – the goal is of course to maximize the success and minimize the costs. The PARADISE framework takes into account lots of things when calculating the user satisfaction, not only things that have to do with robustness. What is most interesting from a robustness point of view is the counting of rejects, cancels, timeouts and means recognition score. Other things taken into account are number of turns, requests for help, barge-ins, elapsed time etc. Rejects are the number of times that the recogniser cannot produce a result with enough confidence. Timeouts are the number of times that the system didn't get a response from the user within reasonable time. Cancels are the number of times that the system interpreted the user input as a request to take one step backwards in the dialog. Mean recognition score need no further explanation (Walker et al., 1999).

Some of these values don't seem to have anything to do with robustness - for instance the number of cancel requests. We believe, however, that every time a user must explicitly tell the system that it isn't behaving the way that the user expected, it is a sign that the system wasn't capable of producing the correct result (compare this to the discussion of the dictionary article in chapter ג). A timeout may be evidence for the same thing.

7 Summary and discussion

One of the most important robustness features, that we have mentioned several times in this paper, are the system’s ability to understand, and prevent, unrealistic expectations of the system. This includes both overestimating the capacity of the system as well as posing questions outside the system's domain. This is important in terms of user satisfaction. The question that we raised when we thought about these remarks in terms of robustness was; does the user expect more from a system that are more personified compared to one that is more static? In studies concerning the August system, se chapter 4.1, it was documented that people came up to the dialog system and tried to just have a friendly conversation.

There are several separate parts in dialog systems that can be improved to make them more robust. One difficult task is to make ASR (automatic speech recognition) better. ASR has serious problems with noisy environments, and dialog systems are quit often employed in environments where the noise level is difficult to control. The robustness requirements on the other components in the dialog system increase tremendously when the speech recognising part fails to give reliable analyses. The task that the ASR are set out to do is getting even harder if the dialog system is meant to be speaker independent. These demands, forces the systems to improve their level of performance so it gets closer the level of what humans perform when it comes to speech recognition.

Another capacity that humans have, that dialog systems are missing in great degree, is the ability to realize when the dialog isn’t going where its suppose to. Humans also take their dialog partner into consideration and they adapt to each other. Both the ability to recover a dialog from misunderstandings and the ability to adapt to the conversation partner are virtues that dialog systems can improve to make their performance more robust.

A problem with defining robustness is the system limits and what is defined as the input of the system. If the dialog system limit is the microphone, a smaller part of the problems for dialog systems can be considered robustness problems than if the system limit is the user’s mouth.

If a system is robust enough depends on what its usage is and for whom it is intended. The more flexibility the system has, the harder the requirements get on almost every component. In the end it is the users that shall be satisfied and they shall feel that they have received the information they expected to get from the system.

One of the conclusions that can be drawn is that it is impossible to talk about the performance or the usability of a dialog system without talking about robustness. One has to define what is a ”sufficiently distorted input” and what is a ”small deviation from a predefined ideal”. If the goal is that the system should be able to interact with humans in an informal and human-like way, the predefined ideal is huge, and most things that lead to poor performance of dialog systems today should be regarded as robustness deficits.

References

Allwood, Jens, Traum David and Jokinen, Kristiina (2000), *Cooperation, dialog and ethics*, International Journal of Human Computer Studies vol. 53, No 6. Academic Press

H. Aust and N. Lenke (1997). *The Philips Dialog System - Applications and Improvements*. In Proc. of the COST Workshop on Speech Technology in the Public Telephone Network: *Where are we today?*, pages 25-32, Rhodes, Greece.

Bell, Linda and Gustafson, Joakim (1999), *Utterance types in the August database*. The Third Swedish Symposium on Multimodal Communication. Linköping Univ., Oct 15-16, 1999

Bell, Linda and Gustafson, Joakim (1999) *Interacting with an animated agent: an analysis of a Swedish database of spontaneous computer directed speech*, In Proc of Eurospeech '99, pages 1143-1146, Budapest, Hungary

Carlson, Rolf and Granström, Björn (1995), *The Waxholm spoken dialog system,* In: Palková Z, ed. Phonetica Pragensia IX. Charisteria viro doctissimo Premysl Janota oblata. Acta Uni-versitatis Carolinae Philologica 1, 1996. Prague: Charles University; 39-52.

Carlson, Rolf and Granström, Björn (1997), *Speech synthesis*, The handbook of phonetic science, Blackwell Publishers Ltd, Oxford.

Danieli, M. and Gerbino, E. (1995), *Metrics for evaluating dialog strategies in a spoken language system.* In Proc. of the AAAI Spring Symposium on Empirical Methods in Discourse Interpretation and Generation, pages 34#39.

Glass, James R. (1999), *Challenges for spoken dialog systems,* Spoken Language Systems Group, MIT Laboratory for computer science, Cambridge.

Granstöm, Björn (1999), *Multi-modal Speech Synthesis with Applications*, Proceedings of the 3rd International School on Neural Nets “Eduardo R. Caianiello” Eds G. Chollet, M. G. Di Benedetto, A. Esposito, M. Marinaro, Springer London.

Lippmann, Richard P. (1997), *Speech recognition by machines and humans*, Speech communication, Elsevier Science B.V.

Lüth, Tim C., Längle, Thomas, Herzog, Gerd, Stopp, Eva, Rembold, Ulrich, *KANTRA: Human-Machine Interaction for Intelligent Robots Using Natural Language,* In 3rd IEEE Int. Workshop on Robot and Human Communication, RO-MAN'94, pp. 106-111, Nagoya, Japan, 1994

McTear, Michael F (2001), *Spoken Dialog Technology: Enabling the Conversational User Interface.* Submitted to ACM Computing Surveys.

Wahlster, Wolfgang (2000) *Mobile Speech-to-Speech Translation of Spontaneous Dialogs: An Overview of the Final Verbmobil System*, In Wahlster, W. (ed.) Verbmobil: Foundations of Speech-to-Speech Translation. Berlin, Heidelberg, New York: Springer pp. 3-2.

Walker, Marilyn A., Kamm, Candace A. and Litman, Diane J. (1998) *Towards Developing General Models of Usability with PARADISE*, In Natural Language Engineering, to appear.