EVALUATING DIALOG SYSTEMS USED IN THE REAL WORLD

Harald Aust[*], *Hermann Ney*[**]

[*]Philips Speech Processing
Weisshausstr. 2
D-52066 Aachen
Germany
aust@pfa.research.philips.com

[**]Lehrstuhl für Informatik VI
RWTH Aachen – University of Technology
D-52056 Aachen
Germany
ney@informatik.rwth-aachen.de

ABSTRACT

An important aspect of creating high performance natural language dialog systems is the question of how they are evaluated. While a universally accepted method for doing so for pure speech recognition exists, this is not clear for speech understanding or dialog systems. We describe the methods we typically use for our systems and argue that it is not sufficient to evaluate their constituents separately. Instead, a measure for a system in its entity is needed.

1. THE PHILIPS DIALOG SYSTEM

The Philips dialog system, which is described in more detail in [2] and [3], is a generic system that allows the development of natural-language inquiry systems, i.e. systems that people can call in order to obtain certain information, or to have a transaction performed. Communication with the system takes place in unrestricted, fluent speech, very similar to the communication with human operators.

In order to allow for easy maintenance and improvements as well as little computational overhead during run-time, we chose a rather simple architecture for our system (Fig. 1): The main components speech recognition, speech understanding, dialog control, and speech output are realized as separate, independent modules that are executed sequentially. As interface between recognition and understanding, *word graphs* are used. This results in a de facto-integration of these components because their combined knowledge can be employed to determine the best path through the word graph which is then taken as the result of the entire process.

In natural-language dialog systems, well-formed, grammatically correct sentences are the exception rather than the rule. In addition, recognition errors tend to corrupt the spoken input considerably. These are the main reasons why our speech understanding component, which was first described in [1], does not try to parse an entire sentence but is looking only for those words and word sequences that carry a meaning with respect to the application – the so-called *concepts*. We use an attributed stochastic context-free grammar that serves three purposes:

- It is used as a language model,
- it identifies the concept instances in the word graph, and
- it computes their meaning.

Concept bigrams and a *filler model* are used in addition to the grammar to find the best path through the incoming word graph. The concept instances along this path are collected and form the meaning of the sentence.

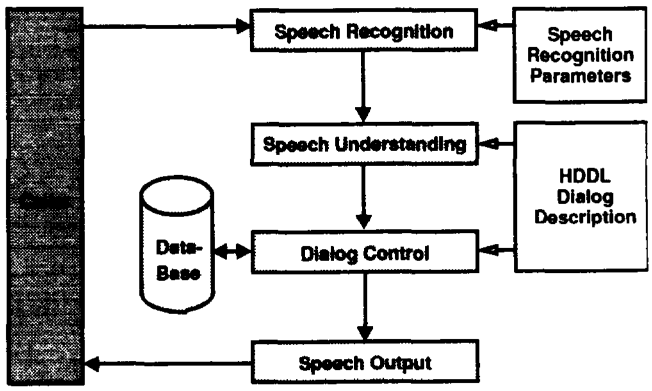

Figure 1: The Architecture of the Philips System

All words that are not part of concepts are ignored during this process. Note, however, that the understanding rate may be improved if *meaningless concepts* are introduced which can, for example, cover phrases like "good morning" or "I'd like to".

The declarative *dialog description language HDDL*, which has been developed specifically for natural-language inquiry systems, is used for the specification of the grammar rules and the entire dialog flow. It is independent of a particular application, database, or language.

2. REALIZED APPLICATIONS

Our system has been used for the realization of many applications from a variety of application areas, in different languages, complex or simple, for internal purposes or publicly available. The most important ones, which are somewhat more extensively described in [4], are the following (status information given is as of October 1997):

- Probably best known is our train timetable inquiry system in the German language, which has been introduced in February 1994 and was, as far as we know, the world's first natural-language dialog system available to the general public. It can provide information on connections between about 1000 German cities. For a detailed description of the system and the experiences from its operation, see [3].
- A similar system, also in German, has been implemented for Swiss Railways (SBB), and has been operational since June 1996. SBB did an extensive survey on this system early in 1997 [5]. 93% of the calls that fell into its domain were completed successfully; in addition, more than 80% of the callers asked expressed satisfaction and will use the system again.
- Another information system has been developed for Lufthansa airways. Its language is German; however, it allows the pronunciation of foreign city names in their respective language (English, French, Spanish etc.). This system is part of a larger, menu-structured installation; a caller is connected to it at a certain point in the hierarchical dialog.
- The same basic multilingual approach is employed in a weather information system that is currently in the testing phase.
- A restaurant guide for the Boston area, in American English, has become our American demonstration system. A caller can specify the desired type of food (e.g. Italian or French), the preferred location, and a price range.
- *SpeechAttendant* is, as the name implies, an automatic telephone attendant. Callers can ask, as usual in fluent speech, for information on up to several hundred employees of a company, e.g. telephone or fax number, office number, e-mail address etc., and can be connected to a desired extension. A version of it has been widely used within Philips Research for more than two years.
- *Bank42* is an internal evaluation system that implements the front end of a banking application. It allows the selection of the language (English or German), and the input of arbitrary-length account and identification numbers, possibly by grouping several digits.

In all these applications, the incoming calls have been recorded and transcribed. While this requires considerable effort, it allows us to evaluate the systems and to obtain training material both for recognition and understanding.

3. EVALUATION

Just as we decided on a system architecture in which the main components are separate, independent modules in order to make modifications and maintenance easier, we would, of course, also like to evaluate each of these constituents individually for very similar reasons. Unfortunately, we will see that this is not really possible, or, to the very least, not overly helpful.

3.1 Evaluation of Speech Recognition

For pure speech recognition systems, the *word error rate*, comprising insertions, substitutions, and deletions of spoken words compared to a written reference sentence, is universally accepted. Of course, this measure cannot be used for the evaluation of speech understanding modules where we are not interested in the accuracy of the word sequence but of the meaning of the sentence, with respect to the application, as it was understood by the system.

3.2 Evaluation of Speech Understanding

In analogy to the speech understanding method of our automatic inquiry systems, we regard the meaning of a sentence as a set of slot-and-filler pairs. It is then straightforward to define an error measure similar to the word error rate in recognition, namely the *slot error rate*, sometimes also called *attribute error rate*. It describes the percentage of slot values inserted, deleted, or confused by the system. Note that this approach deviates considerably from the ATIS evaluation methodology in which not the result from the understanding process is measured directly but the response offered by the system as a reaction to the user input [9]. Since our system does not normally perform a database query after every user utterance, we could not have employed the ATIS method easily.

Similarly to speech recognition systems, we need a reference with which to compare our system's results, in particular if we want the computation of the error rate to be done automatically. This reference can, of course, not be a written sentence but has to be the meaning of it. While the former is available as the result of the transcription process, this is not normally the case for the meaning. But fortunately, we have a means to compute it automatically: We use our speech understanding module to process the written sentence and take the result as the reference.

It is obvious that the reference generated this way will only be reliable if our understanding component is absolutely accurate on written input, i.e. it is able to always determine the correct meaning of a written sentence. While it will probably not be possible to achieve this goal in general, and much less, to prove that a speech understanding component meets this requirement, it has turned out in all of our applications that on realistic data, i.e. data obtained from real users instead of people trying to find the system's limitations, our system shows indeed this performance, namely a slot error rate of 0%, or at least very close to 0%, on written input. This result may be astonishing at first glance; however, the explanation is quite simple: Real users come up with many different sentences to express their desire, most of them grammatically incorrect, but they are using amazingly little variance for actually conveying the meaning. For example, in the train timetable inquiry system in German, we hardly ever observed a phrase for a destination city other than the word "nach" ("to"), followed by a city name. After operating a system for some time, looking at caller utterances, and modifying the grammar accordingly, the system will sooner or later contain all relevant phrases.

Other than giving us a means for automatically creating a reference for speech understanding evaluation, the very high accuracy of our understanding method on written input has another consequence: It does not make sense to evaluate it stand-alone, but only in connection with a recognizer. This can, of course, be done off-line if the word graphs created by the recognizer are available.

Slot error rates must therefore always be seen in relation to the accuracy of the output of the recognizer. Since our recognizer creates word graphs instead of a single sentence, the word error rate obtained on the best sentence, possibly with an adequate language model, is not sufficient: We also need information whether the full meaning of a spoken sentence is contained in a word graph in the first place. If this is not the case, even a perfect understanding method will fail. We can call this measure the *word graph slot error rate*. For evaluations of the speech understanding module alone, only those word graphs where this error rate is 0 should be used. A more sophisticated measure would also tell us how good the best path that contains all information actually is in relation to alternative paths through the graph, but such a measure has yet to be devised.

Note that even in situations in which the understanding component may be faulty already on written input, such an evaluation alone would not be sufficient since it would not tell us anything about the understanding module's robustness towards recognition errors.

Also note that despite the close connection between speech recognition and speech understanding, the slot error rate has no direct relation to the word error rate of the recognizer; in particular, one cannot be computed from the other without additional information. For example, if the recognition fails completely on the sequence "Hello, good morning, I would like to", this still will not lead to an understanding problem since only meaningless words are affected. And the other way round: If the single word "now" is not recognized correctly, multiple understanding errors will result: for the date, the time, and possibly other slots.

The speech understanding evaluation method described above assumes that the functionality and capability of the understanding component does not change with the state of the dialog. This is currently the case in all our systems. If it were not, the evaluation would need the information in which dialog state the sentence was uttered in addition to the word graph representing the sentence itself. It would be easy to obtain, though: When recording an utterance, the system would simply have to also store a description of the current dialog state.

In practical applications, we observe a hierarchy in the perceived severity of value insertions, substitutions, and deletions. Deletions, if they do not occur too frequently, cause little problems; apparently, callers are used to repeating information because of their experience with similar situations with human counterparts. Substitutions are more serious; they require the caller to correct the wrong value – which is, amazingly enough, apparently not an easy task for all people –, and the system to deal with the correction. In addition, the misunderstanding creates an unfavorable impression of the system. Insertions, finally, are by far the most severe problems and lead inevitably to confusion since the system seems to "assume" something the caller never mentioned.

As a consequence, it is usually a good idea not to use the simple slot error rate but a *weighted slot error rate* that takes these circumstances into account. Depending on the application, it may be reasonable to assign a much higher weight to an insertion than to a deletion, and optimize the system accordingly. Note that in our system the insertion/deletion ratio can be controlled by adjusting the weight of the filler models in relation to the concept grammar.

3.3 Evaluation of the Dialog Control Component

We have seen that the performance of a speech understanding module cannot reasonably be evaluated independent of the recognizer because measurements that do not take recognition errors into account are not very realistic and therefore not really helpful. In the case of the evaluation of the dialog control component, the situation is even worse: Coping with potentially erroneous spoken input, including confirmation that would not be necessary otherwise, can be considered the most important task of dialog control. This means that offline evaluations like simulations with Wizard-of-Oz-scenarios can at most be useful in a very early stage of dialog design and development.

The ultimate evaluation criterion for a dialog system is *user satisfaction* since it comprises all aspects, from reliable recognition and understanding to pleasant speech output, to useful results from the database and real-time behavior. Unfortunately, since this measure is both subjective and not easily obtained, it is hard to quantify, and even harder to compare across different applications.

Like apparently all other research groups, we do not have a really good solution to these problems. When evaluating our dialog systems, we typically employ two measures: The *success rate* that tells us how many callers obtained the information they actually asked for, and a user satisfaction rate derived from the answers received to a questionnaire at the end of a dialog or from externally conducted surveys. Since not all people asked actually answer the questions or give their opinion, it is hard to tell how reliable this last measure really is.

All dialog evaluation criteria used today have a severe weakness in common: If a dialog system is changed, and this includes changes in the speech recognition or speech understanding parts, all data gathered so far cannot be used for its evaluation anymore since it does not normally reflect the new dialog flow. This also means that different dialog systems cannot be compared on the same data – a prerequisite that has proven invaluable in the development of speech recognition systems.

4. ONGOING SYSTEM EXTENSIONS

The primary goals of the ongoing extensions of the system are to increase its functionality and user friendliness. To that purpose, we are investigating the following methods:

- To improve the recognition performance, we are increasing the vocabulary size of the recognition system. At the same time, we are introducing a garbage model for out-of-vocabulary words to limit their effect on the recognition of adjacent words.
- Due to the increase in vocabulary size and to the already mentioned constraint of real-time response, faster methods for the search process and for the calculation of the acoustic log-likelihoods have to be incorporated into the system [6], [7].
- To allow a more efficient dialog, we are studying the use of confidence measures by which the number of confirmation requests during the dialog can be directly controlled. The confidence measures and the dialog cost may be used to define a single global cost measure of the dialog.
- Finally, the language model used in recognition can be significantly improved by making it dependent on the state of the dialog. In other words, the language model should keep track of the history of the current dialog.

Just as in our original development, the German train timetable information system serves as a vehicle to incorporate these extensions. In addition, it is being adapted to other languages such as Dutch and French. Ultimately, to evaluate the possible advantages of these extensions, we will have to carry out extensive on-line tests like the ones mentioned in the previous chapters. The extensions considered here are partly performed within the EU-funded project ARISE [8].

5. CONCLUSION

While it would be desirable to be able to evaluate speech recognition, speech understanding, and dialog control separately and independent from each other, we have seen that doing so will typically not lead to really useful results. The major reason for this is that already recognition problems drastically affect the design and performance of the other components.

Many of the problems we encounter when developing natural-language dialog systems would disappear if we had a perfect recognizer. Unfortunately, such a system will most likely not be available in the foreseeable future. Dealing with recognition errors will therefore continue to be the most important task in the further development of speech understanding and dialog systems. To be most effective in these efforts, we will have to regard our systems as logically integrated in order to be able to compensate for potential weaknesses in one component by specific measures taken in another. While improvements in an individual module should never lead to a decrease of the entire system's performance, there is no direct guarantee that they will increase it, either.

Having a measure for evaluating and comparing complete dialog systems would therefore be extremely helpful. However, such a measure has yet to be devised.

REFERENCES

[1] H. Aust, M. Oerder: "Database Query Generation from Spoken Sentences". 2nd Workshop on Interactive Voice Technology for Telecommunications Applications, Kyoto, Japan, pp. 141–144, Sept.1994.

[2] H. Aust, M. Oerder: "Dialogue Control in Automatic Inquiry Systems". ESCA Workshop on Spoken Dialogue Systems, Aalborg, Denmark, pp. 121–124, June 1995. Also published in: 9th Twente Workshop on Language Technology, Enschede, The Netherlands, pp. 45–49, 1995.

[3] H. Aust et al.: "The Philips Automatic Train Timetable Information System". Speech Communication 17, pp. 249–262, Nov. 1995.

[4] H. Aust, N. Lenke: "The Philips Dialog System – Applications and Improvements". COST Telecom Workshop, Rhodes, Greece, pp. 25–32, Sept. 1997.

[5] J.-C. Peng, R. Flückiger, S. Häfeli: "Prisma Voice Konzept". SBB-internal document, Oct. 1996.

[6] S. Ortmanns, A. Eiden, H. Ney, N. Coenen: "Look-Ahead Techniques for Fast Beam Search". Int. Conf. on Acoustics, Speech and Signal Processing, Munich, Germany, pp. 1783–1786, April 1997.

[7] S. Ortmanns, H. Ney, T. Firzlaff: "Fast Likelihood Computation Methods for Continuous Mixture Densities in Large Vocabulary Speech Recognition". Fifth European Conference on Speech Communication and Technology, Rhodes, Greece, pp. 139–142, Sept. 1997.

[8] "ARISE". Language Engineering – Program & Prospects, European Commission, Telematics Applications Programme, Sector D/12, Luxembourg, pp. 29–30, July 1997.

[9] P.J. Price: "Evaluation of Spoken Language Systems: The ATIS Domain". Speech and Natural Language Workshop, Morgan Kaufmann Publishers, San Mateo, CA, USA, pp. 91–95, June 1990.