CHAT: A Conversational Helper for Automotive Tasks

Fuliang Weng[1] Sebastian Varges[2] Badri Raghunathan[1] Florin Ratiu[2] Heather Pon-Barry[1]
Brian Lathrop[4] Qi Zhang[1] Harry Bratt[3] Tobias Scheideck[1] Kui Xu[1] Matthew Purver[2] Rohit Mishra[4]
Annie Lien[1] M Raya[1] S Peters[2] Y Meng[1] J. Russell[1] L Cavedon[2] E Shriberg[3] H Schmidt[1] R Prieto[4]

[1]Research and Technology Center, Robert Bosch Corp., Palo Alto, California, USA
[2]Center for the Study of Language and Information, Stanford University, California, USA
[3]Speech Technology and Research Lab, SRI International, Menlo Park, California, USA
[4]Electronics Research Lab, Volkswagen of America, Palo Alto, California, USA

{fuliang.weng,badri.raghunathan,tobias.scheideck,madhuri.raya,yao.meng}@rtc.bosch.com
{varges,fratiu,mpurver,peters,lcavedon}@csli.Stanford.edu
{harry,ees}@speech.sri.com
{brian.lathrop,rohit.mishra,ramon.prieto}@vw.com

Abstract

Spoken dialogue interfaces, mostly command-and-control, become more visible in applications where attention needs to be shared with other tasks, such as driving a car. The deployment of the simple dialog systems, instead of more sophisticated ones, is partly because the computing platforms used for such tasks have been less powerful and partly because certain issues from these cognitively challenging tasks have not been well addressed even in the most advanced dialog systems. This paper reports the progress of our research effort in developing a robust, wide-coverage, and cognitive load-sensitive spoken dialog interface called CHAT: Conversational Helper for Automotive Tasks. Our research in the past few years has led to promising results, including high task completion rate, dialog efficiency, and improved user experience.

Index Terms: dialog systems, cognitive load, robustness

1. Introduction

Most current applications of spoken language dialogue systems involve narrowly focused language understanding and simple models of dialogue interaction. Real human dialogue, however, is highly context- and situation- dependent, interactive and collaborative with ill-formed utterances or fragments. Skilled communicators are also very conscious about what, when, and how to speak.

Understanding language and modeling natural dialogue of this form is important in building friendly spoken-language interfaces, but it is particularly critical in settings where the user is focused on external tasks, such as flying a helicopter or driving a car. In such scenarios, users cannot plan their utterances ahead of time: they need to be able to access background information spontaneously, issue instructions that build on the context, and the system must be able to interpret the requests and respond to the users properly.

This paper describes major progress made over the past few years in building such a spoken dialogue system for a variety of applications [12]. Various features have been refined or added to the implemented CHAT system, which include:

- Robustness in the face of imperfect input from human users and Speech Recognition (SR). It addresses issues in utterance end-pointing, speech disfluencies, incomplete references to proper names, and phrase fragments.
- Deep language analysis: the system understands not just the keywords, but subtleties in non-content words, such as "a", "the", and "other".
- Situation- and context-dependent interpretation of user utterances: the system selects the best interpretation from multiple sources and decides a right move for an intended device based on a global optimization.
- Collaborativeness: the user negotiates with the system and finds the best available solution.
- Regulated responses. Information is presented in a load-sensitive way so not to overwhelm the user.
- Dynamic updates: information content may be added or extended dynamically.

Two applications have been evaluated with high task completion rates: one is music, where the user may query song databases and operate an MP3 player; the other is restaurant selection (RS), where the user may select an intended restaurant via negotiation.

The paper is organized as follows. Section 2 describes the overall system architecture. Sections 3, 4, 5, and 6 describe its core modules and their functionalities: the Natural Language Understanding (NLU) module, the dialogue manager (DM) module, the Content Optimization (CO) and Knowledge Management (KM) modules, and the Natural Language Generation (NLG) module. Section 7 gives a brief description of data collection setup as well as evaluation results. Finally, we conclude with a comparison with other work.

2. The dialogue system overview

The CHAT system has kept the same architecture as the one reported in [12], which uses an event-based, message-oriented middleware, a popular paradigm for distributed systems, and is especially convenient for dynamic registration of new components. The major difference from the previous version is

the addition of the prosodic end-pointing module and CO module, and new features at each individual module.

Among the modules in Figure 1, we use the Nuance 8.5 speech recognition engine with class-based ngrams and dynamic grammars, and Nuance Vocalizer 3.0 as the TTS engine. The Prosody module provides context-sensitive end-pointing so that the recognizer does not cut off the utterance immediately when there is a hesitation. In the next few sections, five core components, including the NLU, DM, NLG, CO and KM modules will be detailed.

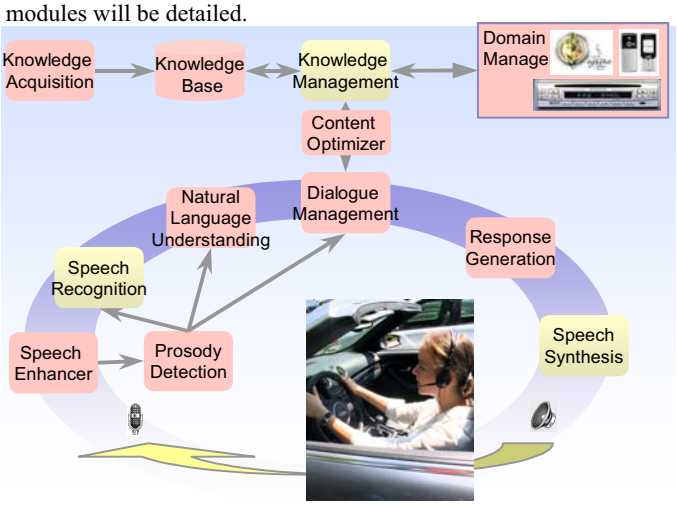

Figure 1 System architecture.

3. The NLU module

The NLU module in our system has been extended to include edit region detection [13], partial proper name identification, and shallow semantic parsing in addition to the original deep structural analysis [12].

One may argue for less sophisticated but robust methods such as using key phrase spotting for understanding so to avoid the trouble of detecting disfluencies or partial proper names (PPN). Our WOZ data shows a strong evidence of PPNs: about 29% of times, people use partial names or their slight variants, instead of full proper names. However, it is difficult to use key phrases to cover variants of partial proper names when they are combined with other phrases. In addition, we intend to use all but the self-edited words for they give us various meanings at different levels. An example for a finer distinction is: *Play a song by Cher* means play *any* song performed by Cher; while *Play the song by Cher* means play a specific song in context.

To provide a backoff solution in case of failed full analysis, we implemented a shallow semantic parser that produces a flat semantic representation similar to [4]. The representation includes dialog acts, actions, and domain-dependent slots (e.g., cuisine type in the RS domain). Some of the attributes are modeled with a maximum entropy approach [14], while others are based on pattern matching. The resulting scores are either the conditional probability or the percentage of overlapping between the best-matched candidate in the database and the actual word sequence in the utterance.

N-best candidates from both deep structural analysis and shallow semantic analysis are used by the later stage DM, which decides the best interpretation together with other choices simultaneously based on the current context.

4. The DM module

The Dialogue Manager (DM) performs pragmatic interpretation of the NLU output (relating the user input to dialogue context and the underlying domain and activity representation) and determines the resulting system activities (dialogue responses, non-verbal actions such as playing songs, or their combinations). We use an information-state-update approach to dialogue management [5] with a rich tree-based representation of dialogue context [6], which includes anaphoric referents and propositional information as well as a structured dialogue move history. Moves are interpreted in context by attachment to an appropriate open parent node (e.g. Answer moves attach to their parent Question nodes), with corresponding rules updating the context.

Importantly, this is linked to, but maintained separately from, the models of devices and activities; these manage the underlying activities, resolving domain referents, monitoring execution, and triggering reports or clarification sub-dialogues as required. System actions are determined by the dialogue move type and the underlying device model: queries or revisions may cause new database queries to be formed and passed to the KM (see below); commands may cause concrete device activities (e.g. playing a song on a MP3 device). The results of these underlying activities then determine system responses to the user: reports of progress or query results; suggestions for refinement or clarification requests in the case of ambiguity or resolution failure. In addition, we provide a separate scripting language to allow easy customization to new domains and devices: this allows the definition of a dialogue move hierarchy, move attachment rules (including possible system output behaviour), and mappings from utterance representations to the moves themselves.

Robustness is increased by allowing multiple possible interpretation methods for move types, and a principled method for choosing between them. Moves can be associated with (possibly underspecified) features of the deep structural output, the shallow semantic output, or their combination. These mappings can be given confidence weights, so that e.g. deep parse information can be preferred in principle, backing off to shallow slot-value pairs if the former is not available, and then to shallow dialogue act or action hypotheses alone. Required arguments for the move (e.g. the NP referent of a MP3 command, or the constraints for a database query) can also be taken from any of these sources, with similar preferences being defined. The overall preferred interpretation (and choice of dialogue move and device) can then be chosen by combining the weights of these mappings with the interpretation confidences provided by the NLU module, and pragmatic factors such as move attachment and domain plausibility (number of database constraints inferred, success of NP resolution etc.). This effectively allows n-best list re-ordering – we do not consider hypotheses in order of NLU confidence alone but on their combined score.

In addition, the close interaction with the underlying activity models [3] means that argument resolution can be revised if necessary: e.g. if a hypothesized constraint value from a deep syntactic parse is preferred but turns out to be incorrect (perhaps leading to null database query results), the DM can back off to hypotheses from the robust semantic classifier and

try again, only being forced to ask the user for clarification if all (reasonably confident) hypotheses fail.

5. The CO and KM modules

The Knowledge Manager (KM) controls the access to knowledge base sources (such as, domain knowledge and device information) and their updates. Domain knowledge is structured according to domain-dependent ontologies, using Protégé, a domain-independent ontology tool.[1] The KM also serves as the repository of device information, such as the device-specific activity model. As a new device is made available, it registers its information with the KM, which then makes it available to the DM.

During a conversation, the DM queries the KM via the CO to get instances matching semantic descriptions derived from utterance. For example, in the MP3 domain, a command to "play some rock music by Cher" results in a query for objects of class *song* with *genre=rock* and *artist=Cher*, where *genre* and *rock* are (inherited) properties of the class *song*.

The CO acts as a content regulator and recommender so that the dialog system will not overwhelm the user nor leave the user in limbo. When many results satisfy the constraints from user, the CO uses the ontology to categorize and output them in a succinct way to reduce the user's cognitive load. At the same time, it may also propose a refinement strategy so that the user can know what to do next. When there is no result from a query, the CO would suggest to the user an alternative constraint based on ontological information and a set of configurable pre-defined strategies, such as relaxing the distance constraint when searching for a restaurant [9].

Additional functionalities from the CO also include:

- Interact with the KM to resolve ambiguous names
- Merge with the previous frame if the current query is a revision instead of a new query
- Provide statistics based on ontological hierarchy.

6. The NLG module

The task of the Natural Language Generation (NLG) module is to verbalize the database query result given a dialogue strategy that has been determined by the dialogue move scripts. However, its role goes well beyond this: NLG is crucial in spoken dialogue systems to provide feedback to the user about what the system understood and what actions it performed as a consequence of its understanding.

The core of the generator is a set of productions written in a production system. We follow the bottom-up generation approach for production systems described in [11] and perform mild overgeneration of candidate moves, followed by ranking. The highest-ranked candidate is selected for output.

Productions map individual database constraints to phrases such as "open for lunch", "within 3 miles" and "a formal dress code", and recursively combine them into NPs. This includes the use of coordination to produce "restaurants with a 5-star rating and a formal dress code", for example. The NPs are integrated into sentence templates, several of which can be combined to form an output candidate turn. For example, a constraint-realizing template "I found no [NP-original] but there are [NUM] [NP-optimized] in my database" (see below for further explanation) can be combined with a follow-up sentence template such as "You could try to look for [NP-constraint-suggestion]".

Ranking of candidate output moves is done by using a combination of factors. First, the ranker computes an alignment score for each candidate, based on its ngram-based overlap with the user utterance. For example, this allows us to prefer "Chinese restaurants" over "restaurants that serve Chinese food" if the user used a wording more similar to the first. Mild overgeneration combined with ranking based on alignment also allows us to map a constraint such as *PriceLevel=0-10* to both "cheap" and "inexpensive", and use alignment to *play back* the original word choice to the user.

Second, ranking uses a variation score to *cycle* over sentence-level paraphrases. In the extreme case of repeated identical user inputs, the system simply chooses one paraphrase after the other, and starts over when all paraphrases have been used.

Third, we use an ngram filter based on bad examples ngrams, removing, for example, "Chinese cheap restaurants" but keeping "cheap Chinese restaurant." For generalization, we replace constraint realizations with semantic tags derived from the constraint names (except for the head noun), for example the trigram "CUISINE PRICE restaurants".

7. Data Collection and Evaluation

WOZ data collection was used to bootstrap the development of the CHAT system [2]. Two batches of experiments have been conducted for MP3 and RS: more than 50 subjects were recruited for performing MP3 or RS while driving in a driving game or a driving simulator. This process gives us insight how human subjects interact with an ideal dialog system, helps us in selecting research topics we need to address, and provides us data for improving the language coverage in both NLU and NLG modules.

The CHAT dialog system is intended for multi-domain applications with wide language coverage, but it is not a system for any general conversation. Careful attention was given to the development of the dialog tasks for the human subjects to perform. Specifically, we try to avoid two undesirable cases:

- Tasks are not constrained enough so that the collected speech becomes irrelevant, unusable, or very sparse.
- Tasks are written in such a way that the subjects tend to use the same language in the task description.

To this end, tasks were written such that the goals of each task were transparent and explicit (to form the intended mental context); however, the language used for communicating these goals to the participant was not explicitly stated (to avoid "copying behavior"). We call this approach a *task-constrained and language-unconstrained* approach. The following example in RS domain illustrates the rational behind this approach:

- Task description: You and a friend would like to have a nice dinner in the ritzy town of Jackson. You both are in the mood for some tasty lobster and crab, complemented with some vanilla-infused drawn butter. Also, the waiters better be on their best behavior, because your friend is somewhat of a snob when it comes to pushing the help around. Find a restaurant that meets your needs.

[1] http://protege.stanford.edu

In the RS domain, 12 criteria (e.g., cuisine type, price level, location) were used across 16 tasks. This is in contrast with the first batch WOZ data collection for the MP3 domain, where much more open-ended scenarios were given to the subjects.

After the system was developed and refined with the WOZ data, evaluation was conducted for both MP3 and RS applications. In the RS evaluation, 9 out of the 16 tasks were selected so that the subjects were able to finish them within one hour. In each task, three criteria are used to pick a restaurant. The nine tasks keep a good balance for the twelve criteria. In MP3 domain, however, we had to design a new set of eleven tasks based on the scenarios. In MP3 evaluation, the number of steps to complete different tasks varies significantly. Thus, two values of Dialog Efficiency were calculated for each subject: Dialog Efficiency Min and Max. Dialog Efficiency Min and Max are defined as the percentage of tasks completed within two turns of the min and max thresholds, respectively. The min threshold is the smallest number of turns ideally required to complete a task. The max threshold is defined to be the highest acceptable number of turns for a task, as determined beforehand based on expert experience with the tasks.

In contrast with MP3 domain, all nine RS tasks require the same number of constraints to be met (i.e., three constraints). Therefore, we report the absolute number of turns used by the subjects. Twenty subjects were recruited independently for each domain.

In the evaluation of the MP3 domain with about 1000 database items, we reached 98% task completion rate, 90% speech recognition accuracy, and 90% semantic accuracy. In addition, 71% is reached for the metric of dialog efficiency min, and 91% for dialog efficiency max. Using the user satisfaction rating system by CU-Communicator [7], we reached a score of 2.24 which is slightly worse than 1.76, a number reached by their latest version, but better than 2.8, an average of their 9 fielded systems. Score 2.24 indicates that users like the system, but not strongly. This could be the effect of the well-designed iPod HMI.

In the evaluation of the RS domain with 2500 restaurants, we reached a task completion rate of 94% with a word recognition rate of 85%, and a semantic accuracy of 89%. On average, the subjects needed 4.1 turns to complete a task. In the current task setup, speaking restaurant names were not encouraged due to poor recognition accuracy for many foreign names in the database. However, our user satisfaction questionnaire showed a mean rating of 2.04, which was not significantly different from the rating of 1.76 reported by [7].

8. Conclusion

Previous research work on conversational dialogue systems has mostly focused on dealing with dialogs that users need to pay full attention to [1]. Their applications include travel planning, flight information, navigation, hotel reservation, and restaurant selection [7,8]. More sophisticated dialogue management research has recently focused on collaborative aspects of human machine dialogues [1,6,10].

While extending the research on the collaborative aspects, our effort specifically focuses on dealing with the conversational phenomena under high stress, including spontaneous speech understanding, maintaining multiple threaded dialogs, robustly selecting best candidates based on global contextual criteria, and load-sensitive information presentation. We are interested in extending and evaluating the system on more challenging domains.

9. Acknowledgements

This project is partially supported by a NIST ATP project, Robert Bosch Corp., and VW of America. Dr. Cavedon has moved to National ICT Australia and RMIT University.

10. References

[1] Allen, J., Byron, D., Dzikovska, M., Ferguson, G., Galescu, L., and Stent, A. Towards Conversational Human-Computer Interaction. *AI Magazine*, 2001.

[2] Cheng, H., Bratt, H., Mishra, R., Shriberg, E., Upson, S., Chen, J., Weng, F., Peters, S., Cavedon, L., and Niekrasz J. A Wizard-of-Oz framework for collecting spoken human computer dialogs. *Proc. of ICSLP-200*0, Jeju, Korea, 2004.

[3] A. Gruenstein and L. Cavedon, Using an activity model to address issues in task-oriented dialogue interaction over extended periods, *AAAI Spring Symposium on Interaction between Humans and Autonomous Systems over Extended Operation*, March 2004.

[4] Y. He and S. Young. A data-driven spoken language understanding system. *IEEE Workshop on Automatic Speech Recognition and Understanding*. 2003.

[5] S. Larsson and D. Traum. Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Natural Language Engineering,* 6(3-4), 2000.

[6] O. Lemon, A. Gruenstein, and S. Peters. Collaborative activities and multi-tasking in dialogue systems. *Traitement Automatique des Langues (TAL),* 43(2), 2002.

[7] B., Pellom, W., Ward, and S., Pradhan. The CU Communicator: An architecture of dialog systems. *Proc. of ICSLP,* Beijing, 2000.

[8] Polifroni, J., Chung, G., and Seneff, S. Towards automatic generation of mixed-initiative dialog systems from web content. *Proc. of Eurospeech*, 2003.

[9] H. Pon-Barry and F. Weng. Evaluation of presentation strategies in a conversational dialog system. *Proc. of ICSLP,* 2006.

[10] Rudnicky, A., Thayer, E., Constantinides, P., Tchou, C., Shern, R., Lenzo, K., Xu, W., and Oh, A. Creating natural dialogs in the Carnegie Mellon Communicator system. *Proc. of Eurospeech*, 1999.

[11] Varge, S. Chart generation using production system. *Proc. of 10th European Workshop on Natural Language Generation*, 2005.

[12] Weng, F., Cavedon, L., Raghunathan, B. Mirkovic, D., Cheng, H., Schmidt, H., Bratt, H., Mishra, R., Peters, S., Upson, S., Shriberg, E., Bergmann, C., Zhao, L. A conversational dialogue system for cognitively overloaded users. *Proc. of ICSLP*, Jeju, Korea, 2004.

[13] Zhang, Q. and Weng, F. Exploring features for identifying edited regions in disfluent sentences. *Proc. of 9th IWPT*, Vancouver, Canada, 2005.

[14] Zhou, Y., Weng, F., Schmidt, H., and Wu, L.D. "A Fast Algorithm for Feature Selection in Conditional Maximum Entropy Modeling". *Proc. of Empirical Methods in Natural Language Processing*, Sapporo, 2003.