An Intelligent Procedure Assistant Built Using REGULUS 2 and ALTERF

Manny Rayner, Beth Ann Hockey, Jim Hieronymus, John Dowding, Greg Aist
Research Institute for Advanced Computer Science (RIACS)
NASA Ames Research Center
Moffet Field, CA 94035
{mrayner,bahockey,jimh,jdowding,aist}@riacs.edu

Susana Early
DeAnza College/NASA Ames Research Center
searly@mail.arc.nasa.gov

Abstract

We will demonstrate the latest version of an ongoing project to create an intelligent procedure assistant for use by astronauts on the International Space Station (ISS). The system functionality includes spoken dialogue control of navigation, coordinated display of the procedure text, display of related pictures, alarms, and recording and playback of voice notes. The demo also exemplifies several interesting component technologies. Speech recognition and language understanding have been developed using the Open Source REGULUS 2 toolkit. This implements an approach to portable grammar-based language modelling in which all models are derived from a single linguistically motivated unification grammar. Domain-specific CFG language models are produced by first specialising the grammar using an automatic corpus-based method, and then compiling the resulting specialised grammars into CFG form. Translation between language centered and domain centered semantic representations is carried out by ALTERF, another Open Source toolkit, which combines rule-based and corpus-based processing in a transparent way.

1 Introduction

Astronauts aboard the ISS spend a great deal of their time performing complex procedures. This often involves having one crew member reading the procedure aloud, while while the other crew member performs the task, an extremely expensive use of astronaut time. The Intelligent Procedure Assistant is designed to provide a cheaper alternative, whereby a voice-controlled system navigates through the procedure under the control of the astronaut performing the task. This project has several challenging features including: starting the project with no transcribed data for the actual target input language, and rapidly changing coverage and functionality. We are using REGULUS 2 and ALTERF to address these challenges. Together, they provide an example-based framework for constructing the portion of the system from recognizer through intepretation that allows us to make rapid changes and take advantage of both rule-base and corpus-based information sources. In this way, we have been able to extract maximum utility out of the small amounts of data initial available to the project and also smoothly adjust as more data has been accumulated in the course of the project.

The following sections describe the procedure assistant application and domain, REGULUS 2 and ALTERF.

2 Application and domain

The system, an early version of which was described in (Aist et al., 2002), is a prototype intelligent voice enabled personal assistant, intended to support astronauts on the International Space Station in carrying out complex procedures. The first production version is tentatively scheduled for introduction some time during 2004. The system reads out each procedure step as it reaches it, using a TTS engine, and also shows the corresponding text and supplementary images in a visual display. Core functionality consists of the following types of commands:

- Navigation: moving to the following step or substep (“next”, “next step”, “next substep”), going back to the preceding step or substep (“previous”, “previous substep”), moving to a named step or substep (“go to step three”, “go to step ten point two”).
- Visiting non-current steps, either to preview future steps or recall past ones (“read step four”, “read note before step nine”). When this functionality is invoked, the non-current step is displayed in a separate window, which is closed on returning to the current step.
- Recording, playing and deleting voice notes (“record voice note”, “play voice note on step three point one”, “delete voice note on substep two”).
- Setting and cancelling alarms (“set alarm for five minutes from now”, “cancel alarm at ten twenty one”).
- Showing or hiding pictures (“show the small waste water bag”, “hide the picture”).
- Changing the TTS volume (“increase/decrease volume”).
- Querying status (“where are we”, “list voice notes”, “list alarms”).
- Undoing and correcting commands (“go back”, “no I said increase volume”, “I meant step four”).

The system consists of a set of modules, written in several different languages, which communicate with each other through the SRI Open Agent Architecture (Martin et al., 1998). Speech recognition is carried out using the Nuance Toolkit (Nuance, 2003).

3 REGULUS 2

REGULUS 2 (Rayner et al., 2003; Regulus, 2003) is an Open Source environment that supports efficient compilation of typed unification grammars into speech recognisers. The basic intent is to provide a set of tools to support rapid prototyping of spoken dialogue applications in situations where little or no corpus data exists. The environment has already been used to build over half a dozen applications with vocabularies of between 100 and 500 words.

The core functionality provided by the REGULUS 2 environment is compilation of typed unification grammars into annotated context-free grammar language models expressed in Nuance Grammar Specification Language (GSL) notation (Nuance, 2003). GSL language models can be converted into runnable speech recognisers by invoking the Nuance Toolkit compiler utility, so the net result is the ability to compile a unification grammar into a speech recogniser.

Experience with grammar-based spoken dialogue systems shows that there is usually a substantial overlap between the structures of grammars for different domains. This is hardly surprising, since they all ultimately have to model general facts about the linguistic structure of English and other natural languages. It is consequently natural to consider strategies which attempt to exploit the overlap between domains by building a single, general grammar valid for a wide variety of applications. A grammar of this kind will probably offer more coverage (and hence lower accuracy) than is desirable for any given specific application. It is however feasible to address the problem using corpus-based techniques which extract a specialised version of the original general grammar.

REGULUS implements a version of the grammar specialisation scheme which extends the Explanation Based Learning method described in (Rayner et al., 2002). There is a general unification grammar, loosely based on the Core Language Engine grammar for English (Pulman, 1992), which has been developed over the course of about ten individual projects. The semantic representations produced by the grammar are in a simplified version of the Core Language Engine’s Quasi Logical Form notation (van Eijck and Moore, 1992).

A grammar built on top of the general grammar is transformed into a specialised Nuance grammar in the following processing stages:

1. The training corpus is converted into a "treebank" of parsed representations. This is done using a left-corner parser representation of the grammar.

2. The treebank is used to produce a specialised grammar in REGULUS format, using the EBL algorithm (van Harmelen and Bundy, 1988; Rayner, 1988).

3. The final specialised grammar is compiled into a Nuance GSL grammar.

4 ALTERF

ALTERF (Rayner and Hockey, 2003) is another Open Source toolkit, whose purpose is to allow a clean combination of rule-based and corpus-driven processing in the semantic interpretation phase. There is typically no corpus data available at the start of a project, but considerable amounts at the end: the intention behind ALTERF is to allow us to shift smoothly from an initial version of the system which is entirely rule-based, to a final version which is largely data-driven.

ALTERF characterises semantic analysis as a task slightly extending the "decision-list" classification algorithm (Yarowsky, 1994; Carter, 2000). We start with a set of *semantic atoms*, each representing a primitive domain concept, and define a semantic representation to be a non-empty set of semantic atoms. For example, in the procedure assistant domain we represent the utterances

please speak up
show me the sample syringe
set an alarm for five minutes from now
no i said go to the next step

respectively as

```
{increase_volume}
{show, sample_syringe}
{set_alarm, 5, minutes}
{correction, next_step}
```

where `increase_volume`, `show`, `sample_syringe`, `set_alarm`, `5`, `minutes`, `correction` and `next_step` are semantic atoms. As well as specifying the permitted semantic atoms themselves, we also define a *target model* which for each atom specifies the other atoms with which it may legitimately combine. Thus here, for example, `correction` may legitimately combine with any atom, but `minutes` may only combine with `correction`, `set_alarm` or a number.[1].

Training data consists of a set of utterances, in either text or speech form, each tagged with its intended semantic representation. We define a set of *feature extraction rules*, each of which associates an utterance with zero or more features. Feature extraction rules can carry out any type of processing. In particular, they may involve performing speech recognition on speech data, parsing on text data, application of hand-coded rules to the results of parsing, or some combination of these. Statistics are then compiled to estimate the probability $p(a \mid f)$ of each semantic atom a given each separate feature f, using the standard formula

$$p(a \mid f) = (N_f^a + 1)/(N_f + 2)$$

where N_f is the number of occurrences in the training data of utterances with feature f, and N_f^a is the number of occurrences of utterances with both feature f and semantic atom a.

The decoding process follows (Yarowsky, 1994) in assuming complete dependence between the features. Note that this is in sharp contrast with the Naive Bayes classifier (Duda et al., 2000), which assumes complete *independence*. Of course, neither assumption can be true in practice; however, as argued in (Carter, 2000), there are good reasons for preferring the dependence alternative as the better option in a situation where there are many features extracted in ways that are likely to overlap.

We are given an utterance u, to which we wish to assign a representation $R(u)$ consisting of a set of semantic atoms, together with a target model comprising a set of rules defining which sets of seman-

[1]The current system post-processes Alterf semantic atom lists to represent domain dependancies between semantic atoms more directly before passing on the result. e.g. `(correction, set_alarm, 5, minutes)` is repackaged as `(correction(set_alarm(time(0,5))))`

tic atoms are consistent. The decoding process proceeds as follows:

1. Initialise $R(u)$ to the empty set.
2. Use the feature extraction rules and the statistics compiled during training to find the set of all triples $\langle f, a, p\rangle$ where f is a feature associated with u, a is a semantic atom, and p is the probability $p(a \mid f)$ estimated by the training process.
3. Order the set of triples by the value of p, with the largest probabilities first. Call the ordered set T.
4. Remove the highest-ranked triple $\langle f, a, p\rangle$ from T. Add a to $R(u)$ iff the following conditions are fulfilled:
    - $p \geq p_t$ for some pre-specified threshold value p_t.
    - Addition of a to $R(u)$ results in a set which is consistent with the target model.
5. Repeat step (4) until T is empty.

Intuitively, the process is very simple. We just walk down the list of possible semantic atoms, starting with the most probable ones, and add them to the semantic representation we are building up when this does not conflict with the consistency rules in the target model. We stop when the atoms suggested are too improbable, that is, they have probabilies below a cut-off threshold.

5 Summary and structure of demo

We have described a non-trivial spoken language dialogue application built using generic Open Source tools that combine rule-based and corpus-driven processing. We intend to demo the system with particular reference to these tools, displaying intermediate results of processing and showing how the coverage can be rapidly reconfigured in an example-based fashion.

References

G. Aist, J. Dowding, B.A. Hockey, and J. Hieronymus. 2002. An intelligent procedure assistant for astronaut training and support. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (demo track)*, Philadelphia, PA.

D. Carter. 2000. Choosing between interpretations. In M. Rayner, D. Carter, P. Bouillon, V. Digalakis, and M. Wirén, editors, *The Spoken Language Translator*. Cambridge University Press.

R.O. Duda, P.E. Hart, and H.G. Stork. 2000. *Pattern Classification*. Wiley, New York.

D. Martin, A. Cheyer, and D. Moran. 1998. Building distributed software systems with the open agent architecture. In *Proceedings of the Third International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology*, Blackpool, Lancashire, UK.

Nuance, 2003. http://www.nuance.com. As of 25 February 2003.

S.G. Pulman. 1992. Syntactic and semantic processing. In H. Alshawi, editor, *The Core Language Engine*, pages 129–148. MIT Press, Cambridge, Massachusetts.

M. Rayner and B.A. Hockey. 2003. Transparent combination of rule-based and data-driven approaches in a speech understanding architecture. In *Proceedings of the 10th EACL*, Budapest, Hungary.

M. Rayner, B.A. Hockey, and J. Dowding. 2002. Grammar specialisation meets language modelling. In *Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP)*, Denver, CO.

M. Rayner, B.A. Hockey, and J. Dowding. 2003. An open source environment for compiling typed unifi cation grammars into speech recognisers. In *Proceedings of the 10th EACL (demo track)*, Budapest, Hungary.

M. Rayner. 1988. Applying explanation-based generalization to natural-language processing. In *Proceedings of the International Conference on Fifth Generation Computer Systems*, pages 1267–1274, Tokyo, Japan.

Regulus, 2003. http://sourceforge.net/projects/regulus/. As of 24 April 2003.

J. van Eijck and R. Moore. 1992. Semantic rules for English. In H. Alshawi, editor, *The Core Language Engine*, pages 83–116. MIT Press.

T. van Harmelen and A. Bundy. 1988. Explanation-based generalization = partial evaluation (research note). *Artificial Intelligence*, 36:401–412.

D. Yarowsky. 1994. Decision lists for lexical ambiguity resolution. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 88–95, Las Cruces, New Mexico.