Context-sensitive speech recognition in ISU dialogue systems: results for the grammar switching approach

Oliver Lemon
School of Informatics
Edinburgh University
Edinburgh EH8 9LW
olemon@inf.ed.ac.uk

Abstract

This research explores how to employ context-sensitive speech recognition in a general way, in the Information-State Update (ISU) approach to dialogue management. The central idea is that different contexts, or dialogue "Information States", can be associated with different language models for speech recognition. In this paper a "grammar switching" approach is presented, based on "active" dialogue move types. It is then shown that this technique leads to more robust speech recognition. An evaluation of a dialogue system using this technique found that 87.9% of recognised utterances were recognised using a context-specific language model, resulting in an 11.5% reduction in the overall utterance recognition error rate, and a 13.4% reduction in concept error rate.

1 Context-sensitive speech recognition in dialogue systems

The basic idea of context-sensitive speech recognition is not new. Finite-state dialogue managers typically define a recognition language model (LM) at each state, and form-based managers often define a LM for each slot, as is commonly done in Voice XML for example. However, this is a laborious and unsystematic process since a designer must anticipate the likely range of user utterances at each point in the dialogue. It also often curtails the freedom of the speaker to say anything at any time in the conversation. In addition, these approaches are domain- and task-specific, and thus are not reusable. The approach presented here is to implement a similar idea more generally and systematically, within a richer (non-finite-state or form-based) model of dialogue context: the Information State Update (ISU) approach (Traum et al., 1999). The general method presented here could be used for a variety of applications, since it only depends on representing the dialogue move types of the user and system, and their dependencies, and not on any application-specific information.

The central idea is to use an "active move list" from dialogue Information States to define a changing search space of language models for speech recognition, to be used whenever the user speaks. We assume that at any point in the dialogue there is a "most active move" of some dialogue move type (for a full description of the system see Lemon and Gruenstein (2004)). In ISU systems generally this is typically the last uttered dialogue move, although there are cases, for example where a clarification subdialogue has just successfully closed, where another dialogue move should be chosen. We then define, for each move type, the name of a language model to be used for speech recognition if that is the type of the most active move. These LMs are defined by dialogue move type, rather than domain-specific slot-value types (e.g. *wh-answer* rather than, say, *city-name*). For instance, if the most active move is a *yes-no-question* then the appropriate language model is defined by a small context-free grammar covering phrases such as "yes", "that's right", "okay", "negative", "maybe", and so on. We call this language model [yn-answer].

In the experimental system, evaluated in the next section, the following LMs were implemented:

- [full]: generated by the whole grammar for the application.
- [wh-answer]: generated by a subgrammar consisting only of "wh-answer" forms such as "the office", "to the school".
- [yn-answer]: generated by a subgrammar consisting only of "yn-answer" forms such as "yeah", "that's right", and so on.
- [alt-answer]: generated by a subgrammar consisting only of "alt-answer" forms such as "now", "later", "do it later", and so on.
- [no-answers]: generated by the whole grammar minus all the "answer" forms.
- [no-corrections-no-wh-answers]: generated by the whole grammar minus "answer" forms and "correction" forms such as "I meant the office", "not the office the lab", and so on.

The dialogue move types were associated with different LMs as shown in the table of Figure 1[1]. This technique is a variant of "conversational games", also known as "dialogue games" (Carlson, 1983), and in the context of task-oriented dialogues, "discourse segments" (Grosz and Sidner, 1986). Such accounts rely on the observation that answers generally follow questions, commands are generally acknowledged, and so on, so that dialogues can be partially described as consisting of "adjacency pairs" of such dialogue moves. A statistical analysis of which dialogue move types typically follow each other could also be used (see e.g. Gabsdil and Lemon (2004)[2].

But what should happen in cases where the user produces an utterance which is not in the coverage of the currently active language model? For example, the currently active LM could be [yn-answers] but the user could produce a command. In cases where recognition fails with the currently active LM, there are several options:

- Reprocess the utterance using the LM related to the next most active move[3].
- Back-off to a "full" LM consisting of all the sentences recognizable for the application, and reprocess the utterance.

Due to the amount of time taken to perform another recognition pass (roughly proportional to the size of the LM) the second strategy is preferable, except for cases where the next most active node has a small associated LM. This is the technique used in the evaluation system. With current processor speeds, both these techniques are feasible. In fact, it is perfectly feasible to run the recognition processes in parallel, as was done in Hockey et al., (2003).

Figure 2 is an excerpt from a Nuance recognizer logfile, showing these dynamic language models in action. Here, the recognizer is in a context where it is using the LM [no-answers] (for instance after just uttering a report) but cannot recognize the user input ("maybe") which is an answer to an earlier system question (e.g. "is this the right car?"). So the system backs-off to the LM [full], and then succeeds in recognizing the answer[4]. Then another user utterance arrives for recognition. In this context there are no active commands that could be corrected, and no open questions, so the recognizer uses [no-corrections-no-answers] and successfully recognizes the user command "zoom in on the car".

[1]"Root" is s special dialogue move type, used at the start of a dialogue, and in other contexts where there are no open questions or active commands.

[2]Here a feature "DMBigramFrequency", calculated from a corpus of dialogues with the system, is used to predict recognition performance, in combination with other features.

[3]This can be iterated, or performed to a certain depth of the active move list.

[4]The recognizer uses a new utterance label (65552) because it treats the back-off recognition pass as a new utterance.

DM Type	Language Model
command	[no-answers]
confirmation	[no-answers]
report	[no-answers]
wh-question	[wh-answer]
yn-question	[yn-answer]
alt-question	[alt-answer]
correction	n/a
yn-answer	n/a
wh-answer	n/a
root	[no-corrections-no-answers]

Figure 1: Language Models associated with Dialogue Move Types

```
started utterance 65550 with grammar .UTTERANCE-no-answers
Result #0:     <rejected> (conf: 37, NL conf:  0)

started utterance 65552 with grammar .UTTERANCE-full
Result #0:     maybe (conf: 81, NL conf: 81)

started utterance 65554 with grammar .UTTERANCE-no-corrections-no-answers
Result #0:     zoom in on the car (conf: 50, NL conf: 47)
```

Figure 2: Excerpt from a Nuance Logfile, showing Context-sensitive Speech Recognition

1.1 Defining suitable Language Models

It might be thought that the process of constructing the required multiple language-models is laborious and time-consuming. However, Gemini, SRI's system for developing bi-directional unification grammars (Dowding et al., 1993), makes this process quite simple. Gemini can be used for parsing and generation, and grammars can be compiled to language models for the Nuance speech recognition system. Similar systems (Bos, 2002; Rayner et al., 2003) are also in development.

Every Gemini grammar rule can be given a feature which is the name of the subgrammar (if any) that it belongs to. When the unification grammar is compiled to its context-free version (Dowding et al., 2001), these subgrammars are preserved, and the Nuance language model compilation process also preserves these named language models. This means that all that is required is to define the subgrammars in the top-level unification grammar formalism. A more laborious alternative is to partition the context-free grammar (Nuance GSL in this case) by hand before compilation. Since the partitioning is to be done by dialogue move type, this is still more general and less labour and maintenance-intensive than finite-state or form-based approaches, which mix together task and dialogue representations.

2 Evaluation

The technique described above was implemented in the WITAS dialogue system (Lemon et al., 2002). Seven members of the University community volunteered to use the system to complete a total of 35 tasks. There were both male and female subjects, all in their twenties or thirties. The subjects were given minimal written instruction on how to use the system before the interaction began. They were then asked to use the system to complete five tasks, in which they directed a simulated robot helicopter to move within a city environment. An example task is "There are reports of a fire at the tower. Check it out and fight the fire if you find one. Then fly the helicopter to the warehouse". Each task was given immediately prior to the start of the interaction, in language the system could not process to prevent users from simply reading the tasks aloud to the system. A given task ended when the user indicated to the system that he or she had finished, or they indicated that they had given up on the task. The system was run in open-microphone mode.

With the context-sensitive recognition system in use, subjects' speech was recorded and the system behaviour logged for each of the five tasks. Data was collected regarding task completion time, steps to completion, and speech recognition error rates. All dialogues were recorded, and the Information States logged as HTML files. The data thus consists of 35 tasks, resulting in 362 user turns, and 731 recognised words (as counted using Nuance batch-recognition). Of utterances which were recognised (at all), 87.9% were recognised using a context-specific language model, with the remainder being handled by backing-off to the full language model when recognition with the context-specific language model had failed to produce any result.

Each subject's speech data was then batch-recognized, without access to dialogue context information, using the full language model for the domain (call this the "normal case"), and the resulting statistics and recognition logs were compared to those from the context-sensitive recognition case. The Nuance batch recognition process effectively simulates (for the purposes of determining speech recognition performance) the performance of the system without the context-sensitive recognizer. We used the same recognition parameters in both cases (i.e. beam width, pruning, etc.).

The performance of the context-sensitive recognition system was evaluated in two ways: overall percentage of utterances recognized and concept accuracy of the recognized utterances (see section 2.2).

2.1 Overall recognition performance

The percentage of utterances recognized in the context-sensitive recognition case was 82.4%, while it was 80.2% in the normal case. Using a paired samples t-test this 2.2% difference between the overall utterance recognition rates in the two samples (number of utterances recognized per subject in the context-sensitive case compared with the normal case) was found to be significant $(t = 2.75, df = 6, p < 0.05)$. The reduction in overall recognition error rate was 11.5%.

Note that the context-sensitive system as implemented here cannot actually perform more poorly than the normal case in terms of number of recognized utterances, due to the fact that it backs-off to the full grammar should its first recognition attempt fail. In such cases the context-sensitive system will be slower than the normal system, but it is faster in the cases where the first recognition attempt succeeds[5] (since a smaller, faster LM is used), so a further study is needed to determine the speed/accuracy trade-offs here.

Note that the context-sensitive case can perform more poorly in the sense of "jumping to conclusions" based on a limited language model (see examples below), so we also need to determine the accuracy of the recognized utterances in each case. For this reason we also evaluate the concept accuracy of the system.

2.2 Concept accuracy

Rather than simply knowing that more utterances are recognized using context-sensitive recognition, we wish to know whether they are recognised correctly, and whether they lead to the correct system actions. It might be the case that context-sensitive recognition indeed recognises more utterances, but recognises them incorrectly, possibly harming overall system performance.

There are several important cases here:

- the user's utterance is recognized correctly by the context-sensitive system, but is not recognized at all by the normal system (e.g. Figure 3, rows 1-4),
- the user's utterance is recognized differently in the normal and context-sensitive cases. The recognition hypothesis in the context-sensitive case is correct (or partially correct) but incorrect for the normal case. Furthermore, the recognition hypothesis for the normal case does not give rise to the user's intended effect[6] (e.g. Figure 3, rows 5-7),

[5]As reported above, this was 87.9 % of the recognized utterances.

[6]There are cases where the normal recognizer output is

- the user's utterance is recognized differently in the normal and context-sensitive cases, and the recognized utterance in the context-sensitive case is incorrect and does not give rise to the user's intended effect, whereas the normal recognition hypothesis is correct or partially correct, or
- both recognition hypotheses are only partially correct, or
- both recognition hypotheses are completely incorrect, or
- there are no recognition hypotheses in either case.

An example of the first case is where the context-sensitive system recognized "fly to the tower" using the LM `[no-corrections-no-wh-answers]`, but the normal system rejected the utterance. An example of the second case is where the dialogue system has asked "Shall I fly to the building now or later?" (and so is subsequently using the LM `[alt-answer]` for recognition) and the user replies with "now" – which is correctly recognized using the context-sensitive system, but is recognized as "no" using the normal system, which would lead to an unintended action. An example of the third case is where the user said "forget about the house" and this was incorrectly recognised as "to the pond" by the context-sensitive system (using the LM `[wh-answer]`), but was correctly recognised using the full LM under batch recognition.

We used the concept accuracy measure of Boros et al., (1996) to compare the performance of the context-sensitive system with the normal system, in respect of each system's ability to correctly recognize user utterances. Concept accuracy is closely correlated with word accuracy, but allows that some word errors do not have a semantic effect (see Chotimongkol and Rudnicky (2001) for examples). Concept accuracy for each utterance is given by the following formula:

not absolutely correct in terms of word errors, but still leads to the user's intended effect (e.g. recognizing "yep" when the user said "yes" leads to the same action in this domain). Such word errors do not count against concept accuracy.

$$CA = 100 \left(1 - \frac{SU_s + SU_i + SU_d}{SU}\right) \%$$

– where SU is the total number of semantic units in the reference answer (i.e. the logical form of the utterance were it recognised correctly), and SU_s, SU_i, and SU_d are the number of semantic units that must be substituted, inserted, or deleted respectively, to correct the actual parser output for the recognised utterance.

Examples of cases where the normal system (in the first column) suffers a concept error that the the context-sensitive system (second column) avoids are shown in Figure 3.

The average concept accuracy in the context-sensitive recognition case was 68.9%, while it was 64.1% in the normal case. Using a paired samples t-test this 4.8% difference in concept accuracy between the two samples was found to be significant $(t = 2.58, df = 6, p < 0.05)$. The reduction in concept error rate was 13.4%.

2.3 Related work

SRI's CommandTalk system (Stent et al., 1999) used a related technique which:

> "used a main grammar (for full sentences), and a second grammar that had full sentences plus isolated NPs. If the system asked a question that could be answered with an isolated NP, then the larger grammar would be activated. The idea was that users were not forced to answer the question, since they had the complete sentence grammar available too." (John Dowding, personal communication).

Note that this technique was adopted to handle isolated NPs occurring as wh-answers, because they were not covered in the full CommandTalk grammar. In the WITAS dialogue system, such isolated NPs are legal utterances in the full grammar, so the problem is not how to include them in the context of a wh-question, but how to exclude them when there is no active wh-question.

Recognition with full LM	**Context-sensitive recognition**	**Context-sensitive Language Model**	**Concept Acc % for full LM**
$<$ rejected $>$	that's it	[yn-answer]	0
$<$ rejected $>$	and follow a truck	[no-answers]	0
$<$ rejected $>$	fly to the power station	[no-corrections-no-answers]	0
$<$ rejected $>$	where are you	[no-corrections-no-answers]	0
no	now	[alt-answer]	0
and the tower	and stop	[no-answers]	0
to the tower	go to the tower	[no-corrections-no-answers]	33.3

Figure 3: Examples of Recognition Hypotheses occurring in the Evaluation Study

3 Conclusion

Speech recognition performance in ISU dialogue systems can be improved by the use of context-sensitive recognition, using a grammar-switching approach based on dialogue move types. Both overall recognition error rates and concept error rates are significantly improved (11.5% and 13.4% reductions respectively) using a general technique which is less labour-intensive and easier to maintain than finite-state or form-based approaches, which mix together domain-specific and dialogue-general representations. A key idea is to define grammars and language models at the more abstract level of dialogue move type (e.g. *wh-answer*) rather than using application-specific slot-filler types (e.g. *destination-city*).

Future work will explore more advanced techniques for determining the correct LM to use in a particular dialogue context – for example the use of machine learning methods (Gabsdil and Lemon, 2004). Further investigation of such techniques is planned in the TALK project[7], see e.g. Lemon and Henderson (2004).

Acknowledgements

With thanks to Alex Gruenstein, Laura Hiatt, and Mark Core. This work was partially funded by Scottish Enterprise under the Edinburgh-Stanford Link programme, and partially by the EC IST FP6 project 507802 "TALK". Thanks also to Nuance Communications Inc. for the use of their speech recognition software.

[7]http:///www.talk-project.org

References

M. Boros, W. Eckert, F. Gallwitz, G. Görz, G. Hanrieder, and H. Niemann. 1996. Towards understanding spontaneous speech: Word accuracy vs. concept accuracy. In *Proceedings ICSLP '96*, volume 2, pages 1009–1012, Philadelphia, PA.

Johan Bos. 2002. Compilation of unification grammars with compositional semantics to speech recognition packages. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002)*, pages 106–112.

Lauri Carlson. 1983. *Dialogue Games: An Approach to Discourse Analysis*. D. Reidel.

A. Chotimongkol and A. I. Rudnicky. 2001. N-best speech hypotheses reordering using linear regression. In *Proceedings of European Conference on Speech Communication and Technology (EuroSpeech 2001)*, pages 1829–1832.

John Dowding, Jean Mark Gawron, Doug Appelt, John Bear, Lynn Cherny, Robert Moore, and Douglas Moran. 1993. GEMINI: a natural language system for spoken-language understanding. In *Proc. 31st Annual Meeting of the ACL*.

J. Dowding, B.A. Hockey, M. J. Gawron, and C. Culy. 2001. Practical issues in compiling typed unification grammars for speech recognition. In *Proceedings of the Thirty-Ninth Annual Meeting of the Association for Compuational Linguistics*.

Malte Gabsdil and Oliver Lemon. 2004. Combining acoustic and pragmatic features to predict recognition performance in spoken dialogue systems. In *Proceedings of ACL-04*, page (to appear).

Barbara Grosz and Candace Sidner. 1986. Attentions, intentions, and the structure of discourse. *Computational Linguistics*, 12(3):175–204.

Beth-Ann Hockey, Oliver Lemon, Ellen Campana, Laura Hiatt, Gregory Aist, Jim Hieronymus, Alexander Gruenstein, and John Dowding. 2003. Targeted help for spoken dialogue systems: intelligent feedback improves naive users' performance. In *Proceedings of European Association for Computational Linguistics (EACL 03)*, pages 147 – 154.

Oliver Lemon and Alexander Gruenstein. 2004. Multithreaded context for robust conversational interfaces: context-sensitive speech recognition and interpretation of corrective fragments. *ACM Transactions on Computer-Human Interaction (ACM TOCHI)*. (to appear).

Oliver Lemon and James Henderson. 2004. Machine learning and the Information State Update approach to dialogue management. In *Proceedings of the Joint AMI-PASCAL-IM2-M4 workshop*, page (to appear).

Oliver Lemon, Alexander Gruenstein, and Stanley Peters. 2002. Collaborative activities and multitasking in dialogue systems. *Traitement Automatique des Langues (TAL)*, 43(2):131 – 154. Special Issue on Dialogue.

Manny Rayner, Beth-Ann Hockey, and John Dowding. 2003. An open source environment for compiling typed unification grammars into speech recognisers. In *Proceedings of EACL 2003 (demonstrations)*, pages 223–226.

Amanda Stent, John Dowding, Jean Mark Gawron, Elizabeth Owen Bratt, and Robert Moore. 1999. The CommandTalk spoken dialogue system. In *Proceedings of the Thirty-Seventh Annual Meeting of the ACL*, pages 183–190, University of Maryland, College Park, MD. Association for Computational Linguistics.

David Traum, Johan Bos, Robin Cooper, Staffan Larsson, Ian Lewin, Colin Matheson, and Massimo Poesio. 1999. A Model of Dialogue Moves and Information State Revision. Technical Report D2.1, Trindi Project.