Beyond Class A: A Proposal for Automatic Evaluation of Discourse

Lynette Hirschman, Deborah A. Dahl, Donald P. McKay, Lewis M. Norton, and Marcia C. Linebarger

Unisys Defense Systems
Center for Advanced Information Technology
PO Box 517
Paoli, PA 19301

Introduction

The DARPA Spoken Language community has just completed the first trial evaluation of spontaneous query/response pairs in the Air Travel (ATIS) domain. [1] Our goal has been to find a methodology for evaluating correct responses to user queries. To this end, we agreed, for the first trial evaluation, to constrain the problem in several ways:

Database Application: Constrain the application to a database query application, to ease the burden of a) constructing the back-end, and b) determining correct responses;

Canonical Answer: Constrain answer comparison to a minimal "canonical answer" that imposes the fewest constraints on the form of system response displayed to a user at each site;

Typed Input: Constrain the evaluation to typed input only;

Class A: Constrain the test set to single unambiguous intelligible utterances taken without context that have well-defined database answers ("class A" sentences).

These were reasonable constraints to impose on the first trial evaluation. However, it is clear that we need to loosen these constraints to obtain a more realistic evaluation of spoken language systems. The purpose of this paper is to suggest how we can move beyond evaluation of class A sentences to an evaluation of connected dialogue, including out-of-domain queries.

Analysis of the Training Data

The training data consisted of almost 800 sentences, approximately 60% of which could be evaluated completely independent of context. Of the remaining sentences, approximately half of them (19%) require context, and almost that many do not have a unique database answer (17%). Table 1 shows these figures for the four sets of ATIS training data; note that the total adds up to more than 100% because some sentences belonged to multiple classes. [2]

CLASSIFICATION	#	%
Total Sentences	774	100
Pure Class A	490	63
Context	145	19
Unanswerable	129	17
Ambiguous	42	5
Ungrammatical	31	3

Table 1: Classification of ATIS Training Data

A Modest Proposal

We originally postponed evaluation of non-class A sentences because there was no consensus on automated evaluation techniques for these sentences. We would like here to propose a methodology for both "unanswerable" sentences and for automated evaluation of context-dependent sentences. By capturing these two additional classes in the evaluation, we can evaluate on more than 90% of the data; in addition, we can evaluate entire (well-formed) dialogues, not just isolated query/answer pairs.

Unanswerable Queries

For unanswerable queries, we propose that the system recognize that the query is unanswerable and generate (for evaluation purposes) a canonical answer such as **UNANSWERABLE_QUERY**. This would be scored correct in exactly those cases where the query is in fact unanswerable. The use of a canonical message side-steps the tricky issue of exactly what kind of error message to issue to the user. This solution is proposed in the general spirit of the Canonical Answer Specification [1] which requires only a minimal answer, in order to impose the fewest constraints on the exact nature of the system's answer to the user. This must be distinguished from the use of **NO_ANSWER**, which flags cases where the system does not attempt to formulate a query. The **NO_ANSWER** response allows the system to admit that it doesn't understand something. By contrast, the **UNANSWERABLE_QUERY** answer actually diagnoses the cases where the system understands the query and determines that the query cannot be answered by the database.

[1] This work was supported by DARPA contract N000014-89-C0171, administered by the Office of Naval Research.

[2] This table counts the so-called context-removable sentences as context dependent, because the answer to such sentences changes depending on whether context is used or not.

```
###Q1 Utterance:  What are the flights from Atlanta to Denver on mid-day on the 5th of July?

>>>D1 Display to the User:
 FLT CODE FLT DAY FRM TO   DEPT  ARRV AL  FLT# CLASSES  EQP MEAL STOP DC  DURA
--------- ------- --- --- ----- ----- -- ----- -------- --- ---- ---- -- -----
   102122 1234567 ATL DEN   840   955 DL   445 FYBMQ    757 B       0 N    195
   102123 1234567 ATL DEN   934  1054 EA   821 FYHQK    72S B       0 N    200
   ...


###Q2 Utterance:  Okay, now I would like to find flights going on to San Francisco on Monday
the 9th of July.

*** Q2 needs info from Q1: Leaving from Denver.

>>>D2 Display to the User:
 FLT CODE FLT DAY FRM TO   DEPT  ARRV AL  FLT# CLASSES  EQP MEAL STOP DC  DURA
--------- ------- --- --- ----- ----- -- ----- -------- --- ---- ---- -- -----
   112516 1234567 DEN SFO  1200  1336 UA   343 FYBMQ    D8S L       0 N    156
   112519 12345-7 DEN SFO  1220  1416 CO  1295 FYQHK    733 L       0 N    176
   ...


###Q3 Utterance:  What would be the fare on United 343?

*** Q3 needs information from previous display D2.

>>>D3 Display to the User:
FARE CODE FRM TO  CLASS FA RESTRICT ONE WAY     RND TRIP
--------- --- --- ----- -- -------- ---------- ------------
7100247   DEN SFO F                 $488.00     $976.00 ...


###Q4 Utterance: What about Continental 1295?

*** Q4 needs display from D2 and query from Q3.
```

Figure 1: Using Context to Understand Queries

Capturing the Context

The major obstacle to evaluation of context-dependent sentences is how to provide the context required for understanding the sentences. If each system were able to replicate the context in which the data is collected, it should be possible to evaluate context-dependent queries. This context (which we will call the "canonical context") consists of the query-answer pairs seen by the subject up to that point during data collection. Figure 1 shows the kind of context dependencies that are found in the ATIS corpus.

These examples show how contextual information is used. Query 2 (*... I would like to find flights going on to San Francisco on Monday the 9th of July*) requires the previous query Q1 to determine that the starting point of this leg is Denver. Query 3 (*What would be the fare on United 343?*) refers to an entity mentioned in the *answer* of Query 2, namely United 343. United 343 may well include several legs, flying from Chicago to Denver to San Francisco, for example, with three fares for the different segments (Chicago to Denver, Chicago to San Francisco, and Denver to San Francisco). However, Query 3 depends on context from the previous display to focus only on the fare from Denver to San Francisco. Finally, Query 4 (*What about Continental 1295?*) requires the previous query Q3 *and its context* to establish what is being asked about (fare from Denver to San Francisco); it also refers to an entity mentioned in the display D2 associated with Query 2 (Continental 1295). By building up a context using information from both the query and the answer, it is possible to interpret these queries correctly. This is shown schematically in Figure 2.

Keeping in Synch

In Figure 3, we show an example of what can happen when context is not properly taken into account. This

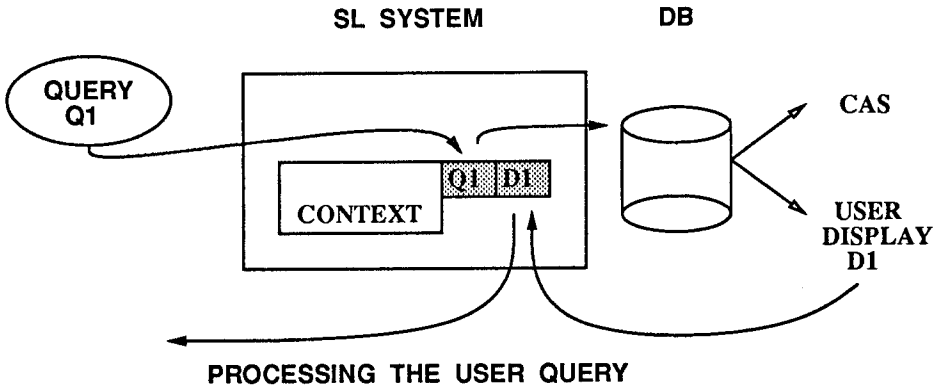

Figure 2: Current Handling of Context in PUNDIT

points out an additional difficulty in evaluating sentences dependent on context, namely the possibility of "getting out of synch". In this example, the system misprocesses the original request, saying that there are no flights from Atlanta to Denver leaving before 11. When the follow-up query asks *Show me the cheapest one*, there is an apparent incoherence, since there is no "cheapest" one in the empty set. However, if the canonical query/answer pairs are provided during evaluation, the system can "resynchronize" to the information originally displayed to the user and thus recognize that it should chose the cheapest flight from the set given in the canonical answer.

Providing the Canonical Context

The above examples illustrate what information is needed in order to understand queries in context. The next question is how to provide this "canonical context" (consisting of the query/answer pairs generated during data collection) for purposes of automated evaluation. Providing the set of queries is, of course, not a problem: this is exactly the set of input data. [3] Providing the canonical answers is more of a problem, because it requires each system to reproduce the answer displayed during data gathering. Since there is no agreement as to what constitutes the best way to display the data, requiring that each system reproduce the original display seems far too constraining. However, we can provide, for evaluation purposes, the display seen by the subject during data collection. The log file in the training data contains this information in human-readable form. It can be provided in more convenient form for automatic processing by representing the display as a list of lists, where the first element in the list is the set of column headings, and the remaining elements are the rows of data. This "canonical display format" is illustrated in Figure 4.

For evaluation, the canonical (transcribed) query and the canonical display would be furnished with each

DISPLAY SHOWN TO USER:

```
 FLT CODE FLT DAY FRM TO   DEPT  ...
--------- ------- --- --- -----
   102122 1234567 ATL DEN   840  ...
   102123 1234567 ATL DEN   934  ...
   ...
```

CANONICAL DISPLAY

```
(
 ('FLT CODE' 'FLT DAY' 'FRM' 'TO' 'DEPT' ... )
 ( 102122     1234567   ATL   DEN  840   ... )
   ...
)
```

Figure 4: Canonical Display Format

query, to provide the full context to the system, allowing it to "resynchronize" at each step in the dialogue. [4] The system could then process the query (which creates any context associated with the query) and answer the query (producing the usual CAS output). It would then reset its context to the state before query processing and add the "canonical context" from the canonical query and from the canonical display, leaving the system with the appropriate context to handle the next query. This is illustrated in Figure 5.

This methodology allows the processing of an entire dialogue, even when the context may not be from the directly preceding query, but from a few queries back. At Unisys, we have already demonstrated the feasibility of substituting an "external" DB answer for the internally generated answer [3]. We currently treat the display (that is, the set of DB tuples returned) as an entity available for reference, in order to capture answer/question dependencies, as illustrated in Figure 3.

[3] Of course, if the input is speech data, then the system could misunderstand the speech data; therefore, to preserve synchronization as much as possible, we propose that the transcribed input be provided for evaluation of speech input.

[4] There is still the possibility that the system misinterprets the query and then needs to use the query as context for a subsequent query. In this case, providing the answer may not help, unless there is some redundancy between the query and the answer.

USER: Show me all flights from Atlanta to Denver leaving before 11.

SYSTEM ANSWER (Wrong):

```
NO INFORMATION SATISFIES YOUR REQUEST
```

CORRECT ANSWER:

```
FLT CODE  FLT DAY FRM TO  DEPT  ARRV  AL FLT#  CLASSES  EQP  MEAL STOP DC DURA
--------- ------- --- --- ----- ----- -- ----- -------- ---  ---- ---- -- ----
102122    1234567 ATL DEN 840    955  DL 445   FYBMQ    757  S/B     0 N  195
102123    1234567 ATL DEN 934   1054  EA 821   FYHQK    72S  S/B     0 N  200
...
```

Follow-up Query:
USER: Show me the cheapest one.

Synchronization lost; can regain with canonical display!

Figure 3: Example of Losing Synchronization

Ambiguous Queries

In addition to the suggestions for handling unanswerable queries and context-dependent queries, there seems to be an emerging consensus that ambiguous queries can be handled by allowing any of several possible answers to be counted as correct. The system would then be resynchronized as described above, to use the canonical answer furnished during data collection.

Evaluation Format

Taking the need for context into consideration and the need to allow systems to resynchronize as much as possible, the proposed form of test input for each utterance in a dialogue is:

- INPUT during TESTING
  - Digitized speech
  - Canonical query for synchronization
  - Canonical display for synchronization
- OUTPUT during TESTING
  - Transcription
  - CAS (with UNANSWERABLE responses)

For evaluation, the system still outputs a transcription and an answer in CAS format; these are evaluated against the SNOR transcription and the reference answer in CAS, as is done now.

With each utterance, the system processes the utterance, then is allowed to "resynchronize" against the correct question-answer pair, provided as part of the evaluation input data before evaluating the next utterance.

Is It Too Easy To Cheat?

One obvious drawback of this proposal is that it makes it extremely easy to cheat – the user is provided with the transcription *and* the database display. It is clearly easy to succumb to the temptation to look at the answer – but it is easy to look at the input sentences under the current system; only honesty prevents us from doing that. Providing a canonical display raises the possibility of deriving the correct answer by a simple reformatting of the canonical display. However, it would be easy to prevent this simple kind of cheating by inserting extra tuples or omitting a required tuple from the canonical display answer. This would make any answer derived from the display not compare correctly to the canonical answer. In short, the issue of cheating does not seem like an insurmountable obstacle: we are now largely on the honor system, and if we wished to make it more difficult to cheat, it is not difficult to think of minor alterations that would protect the system from obvious mappings of input to correct answer.

Evaluating Whole Discourses

There are several arguments in favor of moving beyond class A queries:

- Yield is increased from 60% to over 90%;
- Data categorization is easier (due to elimination of the context-removable class);
- Data validation is easier (no need to rerun context-removable queries);
- Data from different data collection paradigms can be used by multiple sites;
- We address a realistic problem, not just an artificial subset.

This is particularly important in light of the results from the June evaluation. In general, systems performed in the 50-60% range on class A sentences. This means that the coverage of the data was in the 30-40% range.

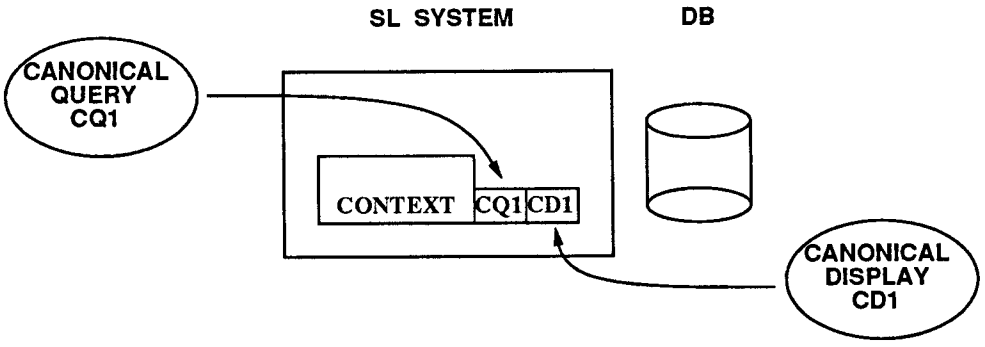

Figure 5: Updating the Context via Canonical Query and Display

If we move on to include unanswerable queries and context dependent queries, we are at least looking at more than 90% of the data. Given that several sites already have the ability to process context-dependent material ([4], [6], [3]), this should enable contractors to report significantly better overall coverage of the corpus.

Subjective Evaluation Criteria

In addition to these fully automated evaluation criteria, we also propose that we include some subjective evaluation criteria, specifically:

- User Satisfaction
- Task Completion Quality and Time

At the previous meeting, the MIT group reported on results using outside evaluators to assess system performance ([5]). We report on a similar experiment at this meeting([2]), in which three evaluators showed good reliability in scoring correct system answers. This indicates that subjective black box evaluation is a feasible approach to system evaluation. Our suggestion is that subjective evaluation techniques be used to supplement and complement the various automated techniques under development.

Conclusion

This proposal does not address several important issues. For example, clearly a useful system would move towards an expert system, and not remain restricted to a DB interface. We agree that this is an important direction, but have not addressed it here. We also agree with observations that the Canonical Answer hides or conflates information. It does not capture the notion of focus, for example. And we have explicitly side-stepped the difficult issues of what kind of detailed error messages a system should provide, how it should handle failed presupposition, how it should respond to queries outside the DB. For the next round, we are suggesting that it is sufficient to recognize the *type* of problem the system has, and to supplement the objective measures with some subjective measures of how actual users react to the system.

References

[1] Sean Boisen, Lance Ramshaw, Damaris Ayuso, and Madeleine Bates. A Proposal for SLS Evaluation In *Proceedings of the DARPA Speech and Natural Language Workshop*, Cape Cod, MA, October 1989.

[2] Deborah A. Dahl, Lynette Hirschman, Lewis M. Norton, Marcia C. Linebarger, David Magerman, Nghi Nguyen, and Catherine N. Ball. Training and evaluation of a language understanding system for a spoken language application. In *Proceedings of the Darpa Speech and Language Workshop*, Hidden Valley, PA, June 1990.

[3] Lewis M. Norton, Deborah A. Dahl, Lynette Hirschman, Marcia C. Linebarger, and Catherine N. Ball. Management and evaluation of interactive dialog in the air travel domain. In *Proceedings of the Darpa Speech and Language Workshop*, Hidden Valley, PA, June 1990.

[4] Wayne Ward. The CMU Air Travel Information Service: Understanding Spontaneous Speech In *Proceedings of the Darpa Speech and Language Workshop*, Hidden Valley, PA, June 1990.

[5] Victor Zue, James Glass, David Goodine, Hong Leung, Michael Phillips, Joseph Polifroni, and Stephanie Seneff. Preliminary evaluation of the voyager spoken language system. In *Proceedings of the DARPA Speech and Natural Language Workshop*, Cape Cod, MA, October 1989.

[6] Victor Zue, James Glass, David Goodine, Hong Leung, Michael Phillips, Joseph Polifroni, and Stephanie Seneff. Preliminary ATIS Development at MIT In *Proceedings of the DARPA Speech and Natural Language Workshop*, Hidden Valley, PA, June, 1990.