Finding Errors Automatically in Semantically Tagged Dialogues

John Aberdeen, Christine Doran, Laurie Damianos,
Samuel Bayer and Lynette Hirschman
The MITRE Corporation
202 Burlington Road
Bedford, MA 01730 USA
+1.781.271.2000
{aberdeen,cdoran,laurie,sam,lynette}@mitre.org

ABSTRACT

We describe a novel method for detecting errors in task-based human-computer (HC) dialogues by automatically deriving them from semantic tags. We examined 27 HC dialogues from the DARPA Communicator air travel domain, comparing user inputs to system responses to look for slot value discrepancies, both automatically and manually. For the automatic method, we labeled the dialogues with semantic tags corresponding to "slots" that would be filled in "frames" in the course of the travel task. We then applied an automatic algorithm to detect errors in the dialogues. The same dialogues were also manually tagged (by a different annotator) to label errors directly. An analysis of the results of the two tagging methods indicates that it may be possible to detect errors automatically in this way, but our method needs further work to reduce the number of false errors detected. Finally, we present a discussion of the differing results from the two tagging methods.

Keywords

Dialogue, Error detection, DARPA Communicator.

1. INTRODUCTION

In studying the contrasts between human-computer (HC) and human-human (HH) dialogues [1] it is clear that many HC dialogues are plagued by disruptive errors that are rarely seen in HH dialogues. A comparison of HC and HH dialogues may help us understand such errors. Conversely, the ability to detect errors in dialogues is critical to understanding the differences between HC and HH communication. Understanding HC errors is also crucial to improving HC interaction, making it more robust, trustworthy and efficient.

The goal of the work described in this paper is to provide an annotation scheme that allows automatic calculation of misunderstandings and repairs, based on semantic information presented at each turn. If we represent a dialogue as a sequence of pairs of partially-filled semantic frames (one for the user's utterances, and one for the user's view of the system state), we can annotate the accumulation and revision of information in the paired frames. We hypothesized that, with such a representation, it would be straightforward to detect when the two views of the dialogue differ (a misunderstanding), where the difference originated (source of error), and when the two views reconverge (correction). This would be beneficial because semantic annotation often is used for independent reasons, such as measurements of concepts per turn [8], information bit rate [9], and currently active concepts [10]. Given this, if our hypothesis is correct, then by viewing semantic annotation as a representation of filling slots in user and system frames, it should be possible to detect errors automatically with little or no additional annotation.

2. SEMANTIC TAGGING

We tagged 27 dialogues from 4 different systems that participated in a data collection conducted by the DARPA Communicator program in the summer of 2000. These are dialogues between paid subjects and spoken language dialogue systems operating in the air travel domain. Each dialogue was labeled with semantic tags by one annotator. We focused on just the surface information available in the dialogues, to minimize inferences made by the annotator.

The semantic tags may be described along two basic dimensions: slot and type. The slot dimension describes the items in a semantic frame that are filled over the course of a dialogue, such as DEPART_CITY and AIRLINE (see Table 1 for the complete list).

The type dimension describes whether the tag is a PROMPT, a FILL, or an OFFER. This type dimension is critical to semantic analysis since it allows one to describe the effect a tag has on slots in the frame. PROMPTs are attempts to gather values to fill slots, e.g., "what city do you want to fly to". FILLs are actual slot fills, e.g., "I'd like to fly to San Francisco". OFFERs represent actual flight information based on previous slot FILLs, e.g., "there is a 9:45 flight to San Francisco on Delta". However, OFFERs often do not exactly match slot FILLs (e.g., the user requests a flight at 9:30, but the closest match flight is at 9:45), and thus must be distinguished from FILLs.

In addition to the two basic dimensions of slot and type, each tag takes a leg attribute to indicate which leg of a trip is being discussed. There is also an initial USER_ID slot which has two types (PROMPT_USER_ID and FILL_USER_ID), but no leg attribute.

Figure 1. Portion of a Sample Dialogue Labeled with Semantic Tags

S0: System said: Welcome to our travel planning system.

<**prompt_user_id**>Please say your PIN number</**prompt_user_id**>.

U1: User said: My PIN is <**fill_user_id**>four four four four</**fill_user_id**>.

S2: System said: What are your travel plans?

U3: User said: I want to go to <**fill_arrive_city leg="1"**>San Francisco</**fill_arrive_city**>.

S4: System said: Traveling to <**fill_arrive_city leg="1"**>San Francisco</**fill_arrive_city**>.

<**prompt_depart_city leg="1"**>Where are you departing from?</**prompt_depart_city**>

U5: User said: <**fill_depart_city leg="1"**>Boston</**fill_depart_city**>.

S6: System said: <**prompt_date leg="1"**>When would you like to leave</**prompt_date**>

<**fill_depart_city leg="1"**>Boston</**fill_depart_city**>?

U7: User said: <**fill_date leg="1"**>July twenty-fifth</**fill_date**> at

<**fill_depart_time leg="1"**>12 noon</**fill_depart_time**>.

S8: System said: I have a <**offer_airline leg="1"**>Delta</**offer_airline**> flight that leaves

<**offer_depart_city leg="1"**>Boston</**offer_depart_city**> at

<**offer_depart_time leg="1"**>11:10AM</**offer_depart_time**> on

<**offer_date leg="1"**>July twenty-fifth</**offer_date**> and arrives in

<**offer_arrive_city leg="1"**>San Francisco</**offer_arrive_city**> at

<**offer_arrive_time**>5:19PM</**offer_arrive_time**>. Is that OK?

Our semantic tag set also includes two special tags, YES and NO, for annotating responses to offers and yes/no questions. Finally, we have two tags, PROMPT_ERASE_ FRAMES and FILL_ERASE_FRAMES, for annotating situations where the frames are erased and the dialogue is restarted (e.g., the user says "start over"). Figure 1 shows part of a sample dialogue with semantic tags. Our semantic tagset is summarized in Table 1.

Table 1. Semantic Tagset

	PROMPT	FILL	OFFER
DEPART_CITY	X	X	X
ARRIVE_CITY	X	X	X
DEPART_AIRPORT	X	X	X
ARRIVE_AIRPORT	X	X	X
DATE	X	X	X
DEPART_TIME	X	X	X
ARRIVE_TIME	X	X	X
AIRLINE	X	X	X
USER_ID	X	X	
ERASE_FRAMES	X	X	
YES	(single bare tag)		
NO	(single bare tag)		

3. ERROR DETECTION

To provide a baseline for comparison to an algorithm that detects errors automatically, we had an annotator (not the same person who did the semantic tagging described above) manually tag the problem areas. This annotator marked four items:

(1) occurrence: where the problem first occurs in the dialogue (e.g. where the user says the item which the system later incorporates incorrectly)

(2) detection: where the user could first be aware that there is a problem (e.g. where the system reveals its mistake)

(3) correction attempt: where the user attempts to repair the error

(4) correction detection: where the user is first able to detect that the repair has succeeded

We next developed an algorithm for automatically finding errors in our semantically tagged dialogues. In this phase of the research, we concentrated on deriving an automatic method for assigning the first two of the four error categories, occurrence and detection (in a later phase we plan to develop automatic methods for correction attempt and correction detection). First, the algorithm derives the turn-by-turn frame states for both the user's utterances and the system's utterances (i.e., what the user heard the system say), paying special attention to confirmation tags such as YES or deletion tags like FILL_ERASE_FRAMES. Then, the algorithm compares patterns of user and system events to hypothesize errors. Occurences and detections are hypothesized for three types of errors: hallucinations (system slot fill without user slot fill), mismatches (system slot fill does not match user slot fill), and prompts after fills (system prompt after user slot fill).

Figure 2 shows a sample dialogue that illustrates several error types. Utterance S12 shows a prompt after fill error – the user has already supplied (in utterance U11) the information the system is requesting. In utterance U13 the user supplies contradictory information, and the system catches this and tries to resolve it in utterances S14 and S16. Next a mismatch error is illustrated – the user specifies ARRIVE_CITY in utterance U17, and the system shows that it has misrecognized

it in utterance S18. The user attempts to correct this misrecognition in utterance U21, and as can be seen from utterance S22, the system again has misrecognized the user's utterance.

Below we describe the results from running the automatic algorithm on our 27 semantically tagged dialogues.

4. RESULTS

In the 27 dialogues considered, a total of 131 items were flagged by one or both of the methods as error items (60 occur, 71 detect). A breakdown of these errors and which method found them is in Table 2.

Table 2. Unique Errors Identified

# errors found by:	**Occur**	**Detect**	**Total**
Both Methods	14	23	37
Automatic Only	28	38	66
Manual Only	18	10	28
Totals	60	71	131

As can be seen in Table 2 the automatic method flagged many more items as errors than the manual method.

Table 3. Error Judgements

	Occur			**Detect**		
	E	**NE**	**Q**	**E**	**NE**	**Q**
Auto	48%	40%	12%	52%	38%	10%
Man	84%	13%	3%	82%	15%	3%

We carefully examined each of the items flagged as errors by the two methods. Three judges (the semantic tagging annotator, the manual error tagging annotator, and a third person who did not participate in the annotation) determined which of the errors found by each of the two methods were real errors (E), not real errors (NE), or questionable (Q). For calculations in the present analysis, we used E as the baseline of real errors, rather than E+Q. Table 3 shows the judgements made for both the automatic and manual method, which are discussed in the next section. It is important to note that human annotators do not perform this task perfectly, with error rates of 13% and 15%. This is also shown in the precision and recall numbers for the two methods in Table 4.

Table 4. Precision & Recall

Precision & Recall	**Occur**		**Detect**	
	P	**R**	**P**	**R**
Automatic	0.48	0.57	0.52	0.84
Manual	0.84	0.77	0.82	0.71

5. ANALYSIS

The automatic method flagged 40 items as errors that the judges determined were not errors (17 occur, 23 detect). These 40 false errors can be classified as follows:

A. 10 were due to bugs in the algorithm or source data

B. 19 were false errors that can be eliminated with non-trivial changes to the semantic tagset and/or algorithm

C. 3 were false errors that could not be eliminated without the ability to make inferences about world knowledge

D. 8 were due to mistakes made by the semantic annotator

One example of the 19 false errors above in B is when the first user utterance in a dialogue is a bare location, it is unclear whether the user intends it to be a departure or arrival location. Our semantic tagset currently has no tags for ambiguous situations such as these. Adding underspecified tags to our tagset (and updating the automatic algorithm appropriately) would solve this problem. Another example is a situation where a system was legitimately asking for clarification about a slot fill, but the algorithm flagged it as prompting for keys that had already been filled. This could be fixed by adding a CLARIFY element to the type dimension (currently PROMPT, FILL, and OFFER). We believe that making these changes would not compromise the generality of our semantic tagset. However, as the point of our approach is to derive errors without much additional annotation, additions to the semantic tagset should only be made when there is substantial justification.

There were also 21 errors (15 occur, 6 detect) that were not detected by the automatic method, but were judged as real errors. These 21 errors may be categorized as follows:

A. 2 were due to bugs in the algorithm

B. 8 were situations where the algorithm correctly flagged the detect point of an error, but missed the associated occur point

C. 6 were situations that could be fixed by modifications to the semantic tagset

D. 1 was an error that could be fixed either by a revision to the semantic tagset or a revision to the algorithm

E. 2 were situations where the system ignored a user fill, and the automatic algorithm interpreted it as no confirmation (not an error). Human judgement is required to detect these errors

F. 2 were due to mistakes made by the semantic annotator

6. PREVIOUS WORK

In Hirschman & Pao [5], annotation was done by manual inspection of the exchanges in the dialogue. Each exchange was evaluated based on the portion of information "visible to the other party". Errors and problems were identified manually

Figure 2. Portion of a Sample Dialogue Illustrating Several Errors

U11: User said: I want to fly from **<fill_depart_city leg="1">**Seattle**</fill_depart_city>** to **<fill_arrive_city leg="1">**Tokyo**</fill_arrive_city>**

S12: System said: **<prompt_arrive_city leg="1">**to which city would you like to fly**</prompt_arrive_city>** from **<fill_depart_city leg="1">**seattle**</fill_depart_city>**

U13: User said: **<fill_arrive_city leg="1">**seattle washington**</fill_arrive_city>**

S14: System said: excuse me, i didn't hear you well. I think you want to fly from **<fill_depart_city leg="1">**seattle**</fill_depart_city>** If this is okay, let's continue. Otherwise, please specify the locations again.

U15: User said: **<yes>**that's okay**</yes>**

S16: System said: **<prompt_arrive_city leg="1">**what city would you like to fly to?**</prompt_arrive_city>**

U17: User said: **<fill_arrive_city leg="1">**tokyo japan**</fill_arrive_city>**

S18: System said: flying to **<fill_arrive_city leg="1">**togiak**</fill_arrive_city>** **<prompt_date leg="1">**on what date?**</prompt_date>**

U19: User said: **<fill_date leg="1">**monday September twenty fifth**</fill_date>**

S20: System said: there don't seem to be any flights **from <fill_depart_city leg="1">**seattle**</fill_depart_city>** to **<fill_arrive_city leg="1">**togiak**</fill_arrive_city>** which depart **<fill_depart_time leg="1">**about ten PM**</fill_depart_time>** on **<fill_date leg="1">**september twenty fifth**</fill_date>**.

U21: User said: destination **<fill_arrive_city leg="1">**tokyo japan**</fill_arrive_city>**

S22: System said: did you say you want to fly to **<prompt_arrive_city leg="1">**san diego**</prompt_arrive_city>**?

and traced back to their point of origin. This is quite similar to our baseline manual annotation described in section 3.

There have been other approaches to detecting and characterizing errors in HC dialogues. Danieli [2] used expectations to model future user utterances, and Levow [6][7] used utterance and pause duration, as well as pitch variability to characterize errors and corrections. Dybkjær, Bernsen & Dybkjær [4] developed a set of principles of cooperative HC dialogue, as well as a taxonomy of errors typed according to which of the principles are violated. Finally, Walker et. al. [11][12] have trained an automatic classifier that identifies and predicts problems in HC dialogues.

7. DISCUSSION

It is clear that our algorithm and semantic tagset, as they stand now, need improvements to reduce the number of false errors detected. However, even now the automatic method offers some advantages over tagging errors manually, the most important of which is that many researchers already annotate their dialogues with semantic tags for other purposes and thus many errors can be detected with no additional annotation. Also, the automatic method associates errors with particular slots, enabling researchers to pinpoint aspects of their dialogue management strategy that need the most work. Finally, Day et. al. [3] have shown that correcting existing annotations is more time efficient than annotating from scratch. In this way, the automatic method may be used to "seed" an annotation effort, with later hand correction.

8. ACKNOWLEDGMENTS

This work was funded by the DARPA Communicator program under contract number DAAB07-99-C201.

9. REFERENCES

[1] Aberdeen, J. and Doran, C. Human-computer and human-human dialogues. *DARPA Communicator Principle Investigators Meeting* (Philadelphia, PA USA 2000). http://www.dsic-web.net/ito/meetings/communicator_sep2000/

[2] Danieli, M. On the use of expectations for detecting and repairing human-machine miscommunication. *Proceedings of AAAI Workshop on Detecting, Repairing and Preventing Human-Machine Miscommunication* (Portland OR, USA 1996).

[3] Day, D., Aberdeen, J., Hirschman, L., Kozierok, R., Robinson, P. and Vilain, M. Mixed-initiative development of language processing systems. In *Proceedings of the Fifth Conference on Applied Natural Language Processing* (Washington DC, USA 1997).

[4] Dybkjær, L., Bernsen, N.O. and Dybkjær, H. Reducing miscommunication in spoken human-machine dialogue. *Proceedings of AAAI Workshop on Detecting, Repairing and Preventing Human-Machine Miscommunication* (Portland OR, USA 1996).

[5] Hirschman, L. and Pao, C. The cost of errors in a spoken language system. *Proceedings of the Third European*

Conference on Speech Communication and Technology (Berlin, Germany 1993).

[6] Levow, G.A. Characterizing and recognizing spoken corrections in human-computer dialogue. *Proceedings of COLING-ACL* (Montreal, Canada 1998).

[7] Levow, G.A. Understanding recognition failures in spoken corrections in human-computer dialogue. *Proceedings of ECSA Workshop on Dialogue and Prosody* (Eindhoven, The Netherlands 1999).

[8] Luo, X. and Papineni, K. IBM DARPA Communicator v1.0. *DARPA Communicator Principle Investigators Meeting* (Philadelphia, PA USA 2000). http://www.dsic-web.net /ito/meetings/communicator_sep2000/

[9] Polifroni, J. and Seneff, S. Galaxy-II as an architecture for spoken dialogue evaluation. *Proceedings of the Second International Conference on Language Resources and Evaluation* (Athens, Greece 2000).

[10] Rudnicky, A. CMU Communicator. *DARPA Communicator Principle Investigators Meeting* (Philadelphia, PA USA 2000). http://www.dsic-web.net/ito/meetings /communicator_sep2000/

[11] Walker, M., Langkilde, I., Wright, J., Gorin, A. and Litman, D. Learning to predict problematic situations in a spoken dialogue system: experiments with how may I help you? *Proceedings of the Seventeenth International Conference on Machine Learning* (Stanford, CA USA 2000).

[12] Walker, M., Wright, J. and Langkilde, I. Using natural language processing and discourse features to identify understanding errors in a spoken dialogue system. *Proceedings of the North American Meeting of the Association of Computational Linguistics* (Seattle, WA USA 2000).