Using Information State to Improve Dialogue Move Identification in a Spoken Dialogue System

Hua Ai[1], *Antonio Roque*[2], *Anton Leuski*[2], *David Traum*[2]

[1]Intelligent Systems Program, University of Pittsburgh, USA
[2]Institute for Creative Technologies, University of Southern California, USA
hua@cs.pitt.edu, {roque,leuski,traum}@ict.usc.edu

Abstract

In this paper we investigate how to improve the performance of a dialogue move and parameter tagger for a task-oriented dialogue system using the information-state approach. We use a corpus of utterances and information states from an implemented system to train and evaluate a tagger, and then evaluate the tagger in an on-line system. Use of information state context is shown to improve performance of the system.
Index Terms: spoken dialogue systems, dialogue management, tagging

10.21437/Interspeech.2007-62

1. Introduction

Context is generally assumed to be important for understanding and participating in dialogue. However, the most successful automated recognition algorithms have tended to focus mainly on features of the input itself. In this paper we examine the use of the information state in an information-state dialogue manager to improve Dialogue Move (DM) and Dialogue Parameter (DP) recognition.

There are a number of different types of context and ways to use context in interpretation tasks. The most obvious type of context (sometimes called "co-text") is the other words surrounding the utterance of interest[1]. The previous interpretation results are also sometimes used (e.g., [1], [2]), to induce a *dialogue model*, comparable to n-gram language models. Another kind of context is the *dialogue state*, which in a finite-state or similar dialogue model, tracks the progress of a dialogue. One common technique (see e.g., [3]) is to have a state-specific speech recognizer or interpreter. In this case, the context is not used explicitly by the recognizer, but is used to choose which model an interpreter will use. There are often state-specific grammars or language models which ignore information from other states.

The *information state approach* to dialogue management [4] involves a flexible approach to representation of context, with theory and domain-specific information updated as the dialogue processes. In this approach, the purpose of an interpretation module is to recognize a set of *dialogue moves* (also sometimes called dialogue acts), which describe the actions taken by the speaker of the utterance, and *dialogue parameters* which specify further information about those actions. Update rules are used to modify the information state when dialogue moves are recognized. There are several ways in which this information state can be used in the interpretation process. Most common for information-state dialogue managers is as a post-processor, where the current information state, as well as the new dialogue moves are used both to decide how to update the information state and how to decide what to say.

Another approach to using context is to provide *expectations* of likely utterances to an interpreter. An interpreter can then merge hypotheses based on the input with hypotheses based on expectations to find a best overall match [5].

We choose a slightly different approach, to treat the information state itself as a set of features that the interpreter can evaluate in the same way it uses input features such as words and n-grams. We test this approach using data from an implemented dialogue system. We compare versions of the interpreter which use context to those which use only the input.

The rest of the paper is structured as follows: section 2 describes the testbed domain and the system. Section 3 describes the experimental method used to test the use of context and section 4 shows our results, and the performance of a context-using interpreter on-line as part of a dialogue system. We conclude in section 5 and provide some directions for future work.

2. Testbed Domain

Our testbed was a corpus of Call for Fire dialogues generated during the evaluation of the Radiobot-CFF spoken dialogue system[6, 7, 8, 9]. Call for Fire (CFF) dialogues are those in which a Forward Observer (FO) specifies a target for artillery to a Fire Direction Center (FDC), according to military protocol. The Radiobot-CFF system plays the role of the FDC in a simulated training environment, where U. S. Army trainees can practice acting as a FO, identifying targets, making calls, and seeing the results in a virtual world.

The system is composed of several components: Automated Speech Recognizer, Interpreter, Dialogue Manager, Generator, and Simulator. The Automated Speech Recognition (ASR) component receives the audio input from trainee and converts it to text format, using the Sonic [10] system. The text is sent to the Interpreter, which identifies the dialogue moves and dialogue parameters in the utterance. The Dialogue Manager uses the text from the ASR component and the dialogue move and dialogue parameter labels from the Interpreter component to update its information state, send commands to the Simulator, and produce content for the Generator to communicate with the user. This paper focuses on an interaction of the Dialogue Manager and the Interpreter: specifically, whether the Interpreter can use information state data from the Dialogue Manager to improve its performance.

The data set we use for this study consists of 1022 FO utterances. Each utterance is transcribed and coded for dialogue moves, dialogue parameters, as well as for the information state

[1]For on-line dialogue systems, only the prior words are available.

Word	steel	one	niner	this	is	gator	niner		one	over
ASR	still	one	niner		is	gator	niner	a	one	over
Quality	subst	correct	correct	deleted	correct	correct	correct	inserted	correct	correct
DM	ID	ID	ID	ID	ID	ID	ID	NULL	ID	none
DP	fdc_id	fdc_id	fdc_id	none	none	fo_id	fo_id	NULL	fo_id	none

Table 1: Tagging on ASR Output

changes that would result from it. Dialogue moves and dialogue parameters are annotated at the word level while information states are annotated at the turn level.

Figure 1 shows the dialogue moves and dialogue parameters for a typical Call for Fire utterance. In this example, the speaker has made an Identification (ID) dialogue move, in which the speaker gives addressee information and a call sign, represented here as the parameters fdc_id and fo_id respectively. The speaker also makes a Warning Order dialogue move requesting a specific kind of artillery fire, here specified by the parameters method_of_fire and method_of_location.

```
Utterance:
  steel one nine this is gator niner one adjust
  fire polar over

Dialogue Move:
   Identification: steel one nine this is gator
                   niner one
Dialogue Parameters:
   fdc_id: steel one nine
   fo_id: gator niner one

Dialogue Move:
   Warning Order: adjust fire polar
Dialogue Parameters:
   method_of_fire: adjust fire
   method_of_location: polar
```

Figure 1: Example Dialogue Moves and Parameters.

Note that a single utterance can contain more than one dialogue move, as well as more than one dialogue parameter. In this domain, dialogue parameters are not exclusively associated with a specific dialogue move: for example, a "target_type" dialogue parameter can be associated with either a "Target Description" dialogue move or an "End of Mission" dialogue move.

3. Approach

Because we worked with a relatively small corpus of 1022 utterances, we ran a set of preliminary experiments to identify which approach best handled the tagging task on this data set. We describe the dialogue move identification task and how we were additionally able to use our tagger to recover from ASR errors. Also, we explored how to identify the most useful information state elements as features for the tagging algorithm.

3.1. DM/DP Identification

Table 1 shows how dialogue move and dialogue parameter identification is handled by word-level tagging. The human speaker utters words such as those shown in the first row, which will later be transcribed by a human for analysis and system training. The output of the ASR component, however, may differ from the words that the speaker actually uttered. As shown in rows two and three, the ASR output may contain words that are inserted, deleted, and substituted. In any case, it is the Interpreter's task to associate each word with a given dialogue move and parameter tag, as shown in rows 4 and 5.

3.2. ASR Error Recovery

It is still an open question in which cases it is better to train such taggers with human-transcribed input (even though one knows the actual input in an implemented system will be noisy because of ASR errors), and when to train them on noisy input (which may compound the problem.)

Testing with noisy input raises some challenges for word-level tagging, in that the set of words in the test set may be different from the transcribed gold standard. For example in Table 1, substitutions such as misrecognizing the first word as "still" instead of "steel", deletions such as the missing the word "this", and insertions such as the word "a", can all cause problems with the DM and DP taggers, as well as with the dialogue manager that will work with this output.

We thus experimented with using a separate CRF tagger to tag the *quality* of each word: substitution, correct, insertion, or deletion. In other words, we tried to identify ASR errors, using a tagger that used the information state context, which the ASR module did not use.

This can be viewed as a way to recover from ASR errors, similar to work such as [12]. This is also similar to work on the recognition level, in which contextual information has been used to choose language models for the speech recognizer [3]. On the understanding level, [11] suggests bringing context information into the task of understanding input speech in the presence of imperfect recognition.

3.3. Tagging Algorithms

To identify the best tagger for our purposes, we compared the performance on our data of Brill's part of speech tagger [13], a J48 Decision Tree implementation by Weka [14], and a Conditional Random Field (CRF) [15, 16] tagger implemented with the Mallet toolkit [17]. Early tests showed the Brill tagger performing at substantially lower accuracies than the other taggers, so we focused on the J48 decision tree and the CRF tagger. As described in section 4.1, the CRF tagger demonstrated the best performance on this corpus.

The CRF algorithm is a variant of the conditional Markov model. It works by using a vector of features indicating whether a particular word occurs in close proximity to the word being tagged. Given a sequence of feature vectors, the CRF tagger produces a sequence of tags. In our experiments we add the information state components to the feature vector. Note that actually two CRF taggers are involved: one for dialogue move tagging and one for dialogue parameter tagging. Future work involves exploring possible interactions between these two taggers.

		Train_Trans:Test_Trans		Train_Trans:Test_ASR		Train_ASR:Test_ASR	
		J48	CRF	J48	CRF	J48	CRF
without context	DM	85.9%	89.4%	76.0%	77.8%	80.4%	84.4%
	DP	90.7%	95.8%	73.6%	77.9%	76.9%	82.1%
	Word	100%	100%	81.0%	81.0%	73.1%	83.1%
with context	DM	86.8%	89.5%	74.7%	78.8%	79.0%	85.6%
	DP	89.9%	96.5%	79.9%	78.4%	75.5%	83.7%
	Word	100%	100%	81.0%	81.0%	72.3%	83.6%

Table 2: Tagging Accuracies in Corpus Evaluation

3.4. Information State Component Selection

The information state tracked by the dialogue manager includes components such as values related to the task (for example, the grid location of a target) and components about the state of the dialogue (for example, the phase of the dialogue: information-gathering, or adjusting). We considered how to identify the information state components most useful to provide as features to the taggers.

Intuitively, the components tracking the state of the dialogue would be the most useful to the taggers. We confirmed this intuition with the feature selection tool provided by Mallet [17], which uses a maximum entropy approach to rank the usefulness of each feature. Of the 27 information state components used by the dialogue manager to complete the domain task, we selected the 9 top features ranked by maximum entropy for use in this experiment.

4. Results

We evaluated our new interpreter in two ways. We evaluated the new Interpreter on the entire corpus to measure its performance improvement compared to the Interpreter that did not use context. Second, we tested in an online system.

4.1. Corpus Evaluation

The results of 4-fold cross-validations on our corpus is shown in Table 2. The first pair of columns show the tagging accuracies of the J48 and CRF taggers when trained on transcriptions and tested on transcriptions. The second pair of columns show their performance when trained on transcriptions and tested on ASR output. The final pair of columns show their performance when trained on ASR output and tested on ASR output. The first set of rows show the accuracies when the taggers were trained without the Information State context as features, and the second set of rows shows the accuracies when the taggers were trained with the Information State context as features.

The CRF tagger outperformed or equalled the J48 tagger on all measures except for the dialogue parameter tagging when trained on transcripts and tested on ASR output. Training and testing on transcription produced the best accuracy measures, but we knew that our online implementation would ultimately need to be tested with ASR output. Of the two CRF taggers tested on ASR output, the one trained on ASR output performed best. We used 2 sided t-test to compare the turn-based prediction accuracies and found that the improvements on DM and DP prediction were statistically significant ($p<0.05$) in both with- and without-context cases. Therefore the CRF tagger trained on ASR output was the one selected for use in the online evaluation.

Table 3 reproduces the accuracy measures of the ASR-trained CRF tagger from Table 2 and adds Reduction in Error Rate when information state context was added as a feature. Dialogue move and dialogue parameter tagging reduction in error rates increased by 7.7 and 8.9% respectively, while word-level ASR correction increased by 2.3%. The improvements gained by adding contextual information are statistically significant ($p<0.05$) for dialogue parameter tagging and show a trend of improvements with 2 sided t-tests ($p<0.08$) for dialogue move and ASR word correction taggings.

	Without Context	With Context	Reduction in Error
DM	84.4%	85.6%	7.7%
DP	82.1%	83.7%	8.9%
Word	83.1%	83.6%	2.3%

Table 3: Corpus Evaluation Tagging Accuracies

	Without Context	With Context	Reduction in Error
DM	89.2%	88.3%	−8.3%
DP	82.3%	88.6%	35.6%
Word	86.5%	91.8%	39.3%

Table 4: Online Evaluation Tagging Accuracies

4.2. Online Evaluation

We then conducted an evaluation of the new Interpreter in an online system. Specifically, we replaced the old Radiobot-CFF tagger with the new tagger, retrained on 100% of the corpus data, and ran an evaluation with 5 users. This resulted in a corpus of 253 utterances, which were later transcribed and annotated in the same way as the original corpus. This evaluation was carried out with a different subject population, and in a different location with different visual guides to locate targets, so the results are not directly comparable to the previous version of the system. However, they do provide a realistic set of data to evaluate the new Interpreter.

After using the transcriptions and annotations to evaluate the performance of the new Interpreter, we also used the ASR output produced during this evaluation to input to the old Interpreter, for comparison. Table 4 shows the results. The new Interpreter performed better on dialogue parameter tagging and on world-level ASR correction, but slightly worse on dialogue move tagging.

4.3. Information State Update in Online Evaluation

Although an improvement in all tagging would have been preferable, the results of the online tagger were not discouraging, because in the current system's Dialogue Manager, the cor-

	Without Context	With Context	Reduction in Error
IS	81.8%	83.8%	11.0%

Table 5: Online Evaluation IS Update Accuracies

rect identification of the existence of a dialogue move is more important than the correct identification of every word's dialogue move tag since the dialogue manager can often recover from errors in dialogue move tagging if the dialogue move is tagged as present in the utterance. The correct identification of every word's dialogue parameter tag, however, is important.

To measure the impact of the new Interpreter on the performance of the Dialogue Manager, we conducted an evaluation on a subset of the online evaluation corpus to measure how well the Dialogue Manager performed its task using both the Interpreter with context and the Interpreter without. This methodology is similar to that in [8]. The information state of the online system was logged after every turn during the online evaluation. After the evaluation, the ASR output was run through the Interpreter without context, and that was then run through the Dialogue Manager to produce a new set of information state output for every turn. These two sets of turns were then compared to a set of information state updates that were generated by a human coder. In this way, the accuracy of the Dialogue Manager in updating its information state components could be measured for each Interpreter.

The results are shown in Table 5. Overall accuracy rate in information state update increases with context, showing a reduction in error rate of 11%. Further research would involve quantifying the relative importance of each of these components.

5. Conclusion and Future Work

We studied the extent to which the components of an information state dialogue manager can be used for dialogue move identification, when used as features of statistical tagger.

Our experiments suggested several strengths of this approach, such as the possibility of using a tagger to recover from ASR errors, as well as some limitations, such as the need for sufficient amounts of training data. Nevertheless, our results so far are promising, showing benefits in our corpus evaluation and in our online evaluation.

Further possibilities for study include an analysis of what kind of user behavior was correctly reacted to by the context-using tagger that the without-context tagger missed, as well as an evaluation in an online system involving a greater number of users. In particular, it would be interesting to compare two such online systems while measuring system-wide evaluation measures such as user satisfaction, task completion, and time-to-task completion.

6. Acknowledgements

This work has been sponsored by the U.S. Army Research, Development, and Engineering Command (RDECOM). Statements and opinions expressed do not necessarily reflect the position or the policy of the United States Government, and no official endorsement should be inferred.

7. References

[1] P. A. Taylor, S. King, S. D. Isard, and H. Wright. 1998. *Intonation and Dialogue Context as Constraints for Speech Recognition*. Language and Speech. Volumn 41, Page 493-512.

[2] M. Poesio and A. Mikheev. 1998. *The Predictive Power of Game Structure in Dialogue Act Recognition: Experimental Results Using Maximum Entropy Estimation*. In Proceedings of ICSLP-98.

[3] D. Bohus and A. Rudnicky. 2003. *RavenClaw: Dialog Management Using Hierarchical Task Decomposition and an Expectation Agenda*. In Proc. Eurospeech 2003, Geneva.

[4] D. Traum and S. Larsson. 2003. *The Information State Approach to Dialogue Management*. Current and New Directions in Discourse and Dialogue. Page 325–353.

[5] Ronnie W. Smith and D. Richard Hipp. 1994. *Spoken Natural Language Dialog Systems: A Practical Approach*. Oxford University Press.

[6] A. Roque and D. Traum. 2006. *An Information State-Based Dialogue Manager for Call for Fire Dialogues*. In Proc. 7th SIGdial Workshop on Discourse and Dialogue.

[7] Susan Robinson, Antonio Roque, Ashish Vaswani, Charles Hernandez, Bill Millspaugh, and David Traum "Evaluation of a Spoken Dialogue System for Virtual Reality Call For Fire Training," Army Science Conference, 2006.

[8] A. Roque, H. Ai, and D. Traum. 2006. *Evaluation of an Information State-Based Dialogue Manager*. In Proc. The 10th Workshop on the Semantics and Pragmatics of Dialogue.

[9] A. Roque, A. Leuski, V. Rangarajan, S. Robinson, A. Vaswani, S. Narayana, and D. Traum. 2006. *Radiobot-CFF: A Spoken Dialogue System for Military Training*. In Proc. Interspeech2006, Pittsburgh.

[10] Bryan Pellom 2001. Sonic: The University of Colorado continuous speech recognizer. Technical Report TRCSLR-2001-01, University of Colorado.

[11] S. LuperFoy and D. Loehr. 1997. *Run-Time Discourse Processing To Supplement Incomplete ASR*. In Proc. IEEE Workshop on ASRU.

[12] Eric K. Ringger. Correcting Speech Recognition Errors. University of Rochester Computer Science Department Technical Report TR-731 and Ph.D. Dissertation.

[13] Brill, Eric 1994. Some Advances in Rule-based Part of Speech Tagging. Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94).

[14] I. H. Witten and E. Frank. 2005. *Data Mining: Practical machine learning tools and techniques*, 2nd Edition. Morgan Kaufmann, San Francisco.

[15] A. McCallum and W. Li. 2003. *Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons*. In Proceedings of The Seventh Conference on Natural Language Learning.

[16] K. Nigam, J. Lafferty, and A. McCallum. 1999. *Using maximum entropy for text classification*. In IJCAI'99 Workshop on Information Filtering.

[17] A. McCallum. 2002. *MALLET: A Machine Learning for Language Toolkit.*. http://mallet.cs.umass.edu. 2002.