Robust Processing of Situated Spoken Dialogue

Pierre Lison and Geert-Jan M. Kruijff

Language Technology Lab
German Research Centre for Artificial Intelligence (DFKI GmbH)
Saarbrücken, Germany

Abstract. Spoken dialogue is notoriously hard to process with standard language processing technologies. Dialogue systems must indeed meet two major challenges. First, natural spoken dialogue is replete with disfluent, partial, elided or ungrammatical utterances. Second, speech recognition remains a highly error-prone task, especially for complex, open-ended domains. We present an integrated approach for addressing these two issues, based on a robust incremental parser. The parser takes word lattices as input and is able to handle ill-formed and misrecognised utterances by selectively relaxing its set of grammatical rules. The choice of the most relevant interpretation is then realised via a discriminative model augmented with contextual information. The approach is fully implemented in a dialogue system for autonomous robots. Evaluation results on a Wizard of Oz test suite demonstrate very significant improvements in accuracy and robustness compared to the baseline.

1 Introduction

Spoken dialogue is one of the most natural means of interaction between a human and a robot. It is, however, notoriously hard to process with standard language processing technologies. Dialogue utterances are often incomplete or ungrammatical, and may contain numerous *disfluencies* like fillers (err, uh, mm), repetitions, self-corrections, fragments, etc. Moreover, even if the utterance is perfectly well-formed and does not contain disfluencies, the dialogue system still needs to accommodate the various *speech recognition errors* thay may arise. This problem is particularly acute for robots operating in real-world environments and dealing with complex, open-ended domains.

Spoken dialogue systems designed for human-robot interaction must therefore be robust to both *ill-formed* and *ill-recognised* inputs. In this paper, we present a new approach to address these two issues. Our starting point is the work done by Zettlemoyer and Collins on parsing using CCG grammars [10]. To account for natural spoken language phenomena (more flexible word order, missing words, etc.), they augment their grammar framework with a small set of non-standard rules, leading to a *relaxation* of the grammatical constraints. A discriminative model over the parses is coupled to the parser, and is responsible for selecting the most likely interpretation(s).

In this paper, we extend their approach in two important ways. First, [10] focused on the treatment of ill-formed input, ignoring the speech recognition issues. Our approach, however, deals with both ill-formed and misrecognized input, in an integrated fashion. This is done by augmenting the set of non-standard rules with new ones specifically tailored to deal with speech recognition errors. Second, we significantly extend the range

of features included in the discriminative model, by incorporating not only *syntactic*, but also *acoustic*, *semantic* and *contextual* information into the model.

An overview of the paper is as follows. We describe in Section 2 the general architecture of our system, and discuss the approach in Section 3. We present the evaluation results on a Wizard-of-Oz test suite in Section 4, and conclude.

2 Architecture

The approach we present in this paper is fully implemented and integrated into a cognitive architecture for autonomous robots (see [4]). It is capable of building up visuospatial models of a dynamic local scene, and of continuously planning and executing manipulation actions on objects within that scene. The robot can discuss objects and their material- and spatial properties for the purpose of visual learning and manipulation tasks. Figure 1 illustrates the architecture for the communication subsystem.

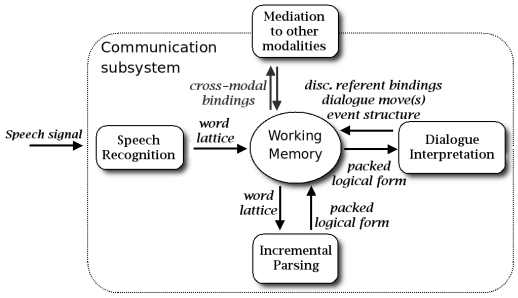

Fig. 1. Architecture schema of the communication subsystem (only for comprehension).

Starting with speech recognition, we process the audio signal to establish a *word lattice* containing statistically ranked hypotheses about word sequences. Subsequently, parsing constructs grammatical analyses for the given word lattice. A grammatical analysis constructs both a syntactic analysis of the utterance, and a representation of its meaning. The analysis is based on an incremental chart parser[1] for Combinatory Categorial Grammar [8]. These meaning representations are ontologically richly sorted, relational structures, formulated in a (propositional) description logic, more precisely in HLDS [1][2]. Finally, at the level of dialogue interpretation, the logical forms are resolved against a dialogue model to establish co-reference and dialogue moves.

3 Approach

3.1 Grammar relaxation

Our approach to robust processing rests on the idea of **grammar relaxation**: the grammatical constraints specified in the grammar are "relaxed" to handle slightly ill-formed

[1] Built using the OpenCCG API: http://openccg.sf.net

[2] An example of such meaning representation (HLDS logical form) is given in Figure 2.

or misrecognised utterances. Practically, the grammar relaxation is done via the introduction of *non-standard CCG rules* [10]. We describe here two families of relaxation rules: the *discourse-level composition rules* and the *ASR correction rules* [5].

Discourse-level composition rules In natural spoken dialogue, we may encounter utterances containing several independent "chunks" without any explicit separation (or only a short pause or a slight change in intonation), such as "*yes take the ball right and now put it in the box*". These chunks can be analysed as distinct "discourse units". Syntactically speaking, a discourse unit can be any type of saturated atomic categories – from a simple discourse marker to a full sentence.

The type-changing rule $\mathbf{T}_{du}$ converts atomic categories into discourse units:

$$\mathsf{A} : @_i f \Rightarrow \mathsf{du} : @_i f \qquad (\mathbf{T}_{du})$$

where A represents an arbitrary saturated atomic category (s, np, pp, etc.).

Rule $\mathbf{T}_C$ then integrates two discourse units into a single structure:

$$\mathsf{du} : @_a x \Rightarrow \mathsf{du} : @_c z \;/\; \mathsf{du} : @_b y \qquad (\mathbf{T}_C)$$

where the formula $@_c z$ is defined as:

$$\begin{aligned} @_{\{c:\text{d-units}\}}(&\mathbf{list} \wedge \\ &(\langle \text{FIRST} \rangle\ a \wedge x) \wedge \\ &(\langle \text{NEXT} \rangle\ b \wedge y)) \end{aligned} \qquad (1)$$

ASR error correction rules Speech recognition is highly error-prone. It is however possible to partially alleviate this problem by inserting error-correction rules (more precisely, new lexical entries) for the most frequently misrecognised words. If we notice for instance that the ASR frequently substitutes the word "wrong" for "round" (because of their phonological proximity), we can introduce a new lexical entry to correct it:

$$round \vdash \mathsf{adj} : @_{attitude}(\mathbf{wrong}) \qquad (2)$$

A small set of new lexical entries of this type have been added to our lexicon to account for the most frequent recognition errors.

3.2 Parse selection

Using more powerful rules to relax the grammatical analysis tends to increase the number of parses. We hence need a mechanism to discriminate among the possible parses. The task of selecting the most likely interpretation among a set of possible ones is called *parse selection*. Once the parses for a given utterance are computed, they are filtered or selected in order to retain only the most likely interpretation(s). This is done via a (discriminative) statistical model covering a large number of features.

Formally, the task is defined as a function $F : \mathcal{X} \rightarrow \mathcal{Y}$ where $\mathcal{X}$ is the set of possible inputs (in our case, $\mathcal{X}$ is the space of *word lattices*), and $\mathcal{Y}$ the set of parses. We assume:

1. A function $\mathbf{GEN}(x)$ which enumerates all possible parses for an input x. In our case, the function represents the admissibles parses of the CCG grammar.
2. A *d*-dimensional feature vector $\mathbf{f}(x, y) \in \Re^d$, representing specific features of the pair (x, y) (for instance, acoustic, syntactic, semantic or contextual features).
3. A parameter vector $\mathbf{w} \in \Re^d$.

The function F, mapping a word lattice to its most likely parse, is then defined as:

$$F(x) = \underset{y \in \mathbf{GEN}(x)}{\operatorname{argmax}} \ \mathbf{w}^T \cdot \mathbf{f}(x, y) \tag{3}$$

where $\mathbf{w}^T \cdot \mathbf{f}(x, y)$ is the inner product $\sum_{s=1}^{d} w_s \ f_s(x, y)$, and can be seen as a measure of the "quality" of the parse. Given the parameter vector $\mathbf{w}$, the optimal parse of a given word lattice x can be therefore easily determined by enumerating all the parses generated by the grammar, extracting their features, computing the inner product $\mathbf{w}^T \cdot \mathbf{f}(x, y)$, and selecting the parse with the highest score.

3.3 Learning

Training data To estimate the parameters $\mathbf{w}$, we need a set of training examples. Since no corpus of situated dialogue adapted to our task domain is available to this day – let alone semantically annotated – we followed the approach advocated in [9] and *generated* a corpus from a hand-written task grammar. We first designed a small grammar covering our task domain, each rule being associated with a HLDS representation and a weight. Once specified, the grammar is then randomly traversed a large number of times, resulting in a large set of utterances along with their semantic representations.

Perceptron learning The algorithm we use to estimate the parameters $\mathbf{w}$ using the training data is a **perceptron**. The algorithm is fully online - it visits each example in turn, in an incremental fashion, and updates $\mathbf{w}$ if necessary. Albeit simple, the algorithm has proven to be very efficient and accurate for the task of parse selection [3,10].

The pseudo-code for the online learning algorithm is detailed in [**Algorithm 1**].

The parameters $\mathbf{w}$ are first initialised to arbitrary values. Then, for each pair (x_i, z_i) in the training set, the algorithm computes the parse y' with the highest score according to the current model. If this parse matches the best parse associated with z_i (which we denote y^*), we move to the next example. Else, we perform a perceptron update on the parameters:

$$\mathbf{w} = \mathbf{w} + \mathbf{f}(x_i, y^*) - \mathbf{f}(x_i, y') \tag{4}$$

The iteration on the training set is repeated T times, or until convergence.

3.4 Features

As we have seen, the parse selection operates by enumerating the possible parses and selecting the one with the highest score according to the linear model parametrised by $\mathbf{w}$. The accuracy of our method crucially relies on the selection of "good" features $\mathbf{f}(x, y)$ for our model - that is, features which help *discriminating* the parses. In our model, the features are of four types: semantic features, syntactic features, contextual features, and speech recognition features.

Algorithm 1 Online perceptron learning

Require: - set of n training examples $\{(x_i, z_i) : i = 1...n\}$
- T: number of iterations over the training set
- $\texttt{GEN}(x)$: function enumerating the parses for an input x according to the grammar.
- $\texttt{GEN}(x, z)$: function enumerating the parses for an input x with semantics z.
- $L(y)$ maps a parse tree y to its logical form.
- Initial parameter vector $\mathbf{w_0}$

% *Initialise*
$\mathbf{w} \leftarrow \mathbf{w_0}$

% *Loop T times on the training examples*
for $t = 1...T$ **do**
  for $i = 1...n$ **do**

    % *Compute best parse according to current model*
    Let $y' = \text{argmax}_{y \in \mathbf{GEN}(x_i)} \mathbf{w}^T \cdot \mathbf{f}(x_i, y)$

    % *If the decoded parse $\neq$ expected parse, update the parameters*
    if $L(y') \neq z_i$ **then**

      % *Search the best parse for utterance x_i with semantics z_i*
      Let $y^* = \text{argmax}_{y \in \mathbf{GEN}(x_i, z_i)} \mathbf{w}^T \cdot \mathbf{f}(x_i, y)$

      % *Update parameter vector* $\mathbf{w}$
      Set $\mathbf{w} = \mathbf{w} + \mathbf{f}(x_i, y^*) - \mathbf{f}(x_i, y')$

    end if

  end for
end for

return parameter vector $\mathbf{w}$

Semantic features Semantic features are defined on *substructures* of the logical form. We define features on the following information sources: the nominals, the ontological sorts of the nominals, and the dependency relations (following [2]). These features help us handle various forms of lexical and syntactic ambiguities.

Syntactic features Syntactic features are features associated to the *derivational history* of a specific parse. The main use of these features is to *penalise* to a correct extent the application of the non-standard rules introduced into the grammar.

To this end, we include in the feature vector $\mathbf{f}(x, y)$ a new feature for each non-standard rule, which counts the number of times the rule was applied in the parse. In the derivation shown in Figure

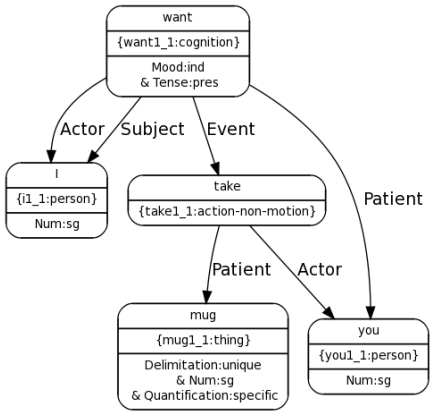

Fig. 2. Logical form for "*I want you to take the mug*".

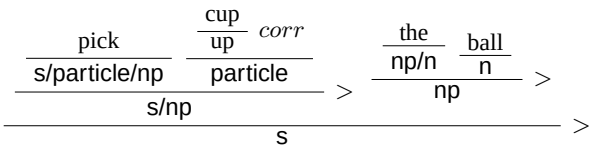

Fig. 3. CCG derivation of *"pick cup the ball"*.

3, the rule $corr$ (application of an ASR correction rule) is applied once, so the corresponding feature value is set to 1. These syntactic features can be seen as a *penalty* given to the parses using these non-standard rules, thereby giving a preference to the "normal" parses over them. This mechanism ensures that the grammar relaxation is only applied "as a last resort" when the usual grammatical analysis fails to provide a full parse.

Contextual features One striking characteristic of spoken dialogue is the importance of context. Understanding the visual and discourse contexts is crucial to resolve potential ambiguities and compute the most likely interpretation(s) of a given utterance. The feature vector f (x, y) therefore includes various contextual features [5]. The dialogue system notably maintains in its working memory a list of contextually activated words [7]. This list is continuously updated as the dialogue and the environment evolves. For each context-dependent word, we include one feature counting its occurrence in the utterance.

Speech recognition features Finally, the feature vector $\mathbf{f}(x, y)$ also includes features related to the *speech recognition*. The ASR module outputs a set of (partial) recognition hypotheses, packed in a word lattice. One example is given in Figure 4. To favour the hypotheses with high confidence scores (which are, according to the ASR statistical models, more likely to reflect what was uttered), we introduce in the feature vector several acoustic features measuring the likelihood of each recognition hypothesis.

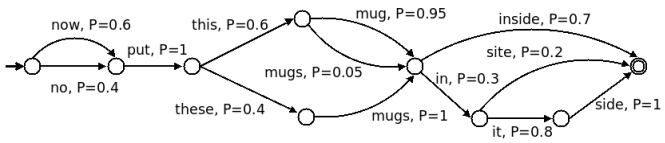

Fig. 4. Example of word lattice

4 Evaluation

We performed a quantitative evaluation of our approach, using its implementation in a fully integrated system (cf. Section 2). To set up the experiments for the evaluation, we have gathered a Wizard-of-Oz corpus of human-robot spoken dialogue for our task-domain, which we segmented and annotated manually with their expected semantic

interpretation. The data set contains 195 individual utterances along with their complete logical forms.

Three types of quantitative results are extracted from the evaluation results: *exact-match*, *partial-match*, and *word error rate*. Tables 1, 2 and 3 illustrate the results, broken down by use of grammar relaxation, use of parse selection, and number of recognition hypotheses considered. Each line in the tables corresponds to a possible configuration. Tables 1 and 2 give the precision, recall and F_1 value for each configuration (respectively for the exact- and partial-match), and Table 3 gives the Word Error Rate [WER].

	Size of word lattice (number of NBests)	Grammar relaxation	Parse selection	Precision	Recall	F_1-value
(Baseline)	1	No	No	40.9	45.2	**43.0**
.	1	No	Yes	59.0	54.3	56.6
.	1	Yes	Yes	52.7	70.8	60.4
.	3	Yes	Yes	55.3	82.9	66.3
.	5	Yes	Yes	55.6	84.0	66.9
(Full approach)	10	Yes	Yes	55.6	84.9	**67.2**

Table 1. Exact-match accuracy results (in percents).

	Size of word lattice (number of NBests)	Grammar relaxation	Parse selection	Precision	Recall	F_1-value
(Baseline)	1	No	No	86.2	56.2	**68.0**
.	1	No	Yes	87.4	56.6	68.7
.	1	Yes	Yes	88.1	76.2	81.7
.	3	Yes	Yes	87.6	85.2	86.4
.	5	Yes	Yes	87.6	86.0	86.8
(Full approach)	10	Yes	Yes	87.7	87.0	**87.3**

Table 2. Partial-match accuracy results (in percents).

The baseline corresponds to the dialogue system with no grammar relaxation, no parse selection, and use of the first NBest recognition hypothesis. Both the partial-, exact-match accuracy results and the WER demonstrate statistically significants improvements over the baseline. We also observe that the inclusion of more ASR recognition hypotheses has a positive impact on the accuracy results.

Size of word lattice (NBests)	Grammar relaxation	Parse selection	WER
1	No	No	**20.5**
1	Yes	Yes	19.4
3	Yes	Yes	16.5
5	Yes	Yes	15.7
10	Yes	Yes	**15.7**

Table 3. Word error rate (in percents).

5 Conclusions

We presented an *integrated* approach to the processing of (situated) spoken dialogue, suited to the specific needs and challenges encountered in human-robot interaction.

In order to handle disfluent, partial, ill-formed or misrecognized utterances, the grammar used by the parser is "relaxed" via the introduction of a set of *non-standard rules* which allow for the combination of discourse fragments or the correction of speech recognition errors. The relaxed parser yields a (potentially large) set of parses, which are then retrieved by the parse selection module. The parse selection is based on a discriminative model exploring a set of relevant semantic, syntactic, contextual and acoustic features extracted for each parse.

The outlined approach is currently being extended in new directions, such as the exploitation of parse selection *during* incremental parsing to improve the parsing efficiency [6], the introduction of more refined contextual features, or the use of more sophisticated learning algorithms, such as Support Vector Machines.

References

1. J. Baldridge and G.-J. M. Kruijff. Coupling CCG and hybrid logic dependency semantics. In *ACL'02: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 319–326, Philadelphia, PA, 2002. Association for Computational Linguistics.
2. S. Clark and J. R. Curran. Log-linear models for wide-coverage ccg parsing. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 97–104, Morristown, NJ, USA, 2003. Association for Computational Linguistics.
3. M. Collins and B. Roark. Incremental parsing with the perceptron algorithm. In *ACL '04: Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, page 111, Morristown, NJ, USA, 2004. Association for Computational Linguistics.
4. N. A. Hawes, A. Sloman, J. Wyatt, M. Zillich, H. Jacobsson, G.-J. M. Kruijff, M. Brenner, G. Berginc, and D. Skocaj. Towards an integrated robot with multiple cognitive functions. In *Proc. AAAI'07*, pages 1548–1553. AAAI Press, 2007.
5. P. Lison. Robust processing of situated spoken dialogue. Master's thesis, Universität des Saarlandes, Saarbrücken, 2008. http://www.dfki.de/∼plison/pubs/thesis/main.thesis.plison2008.pdf.
6. P. Lison. A method to improve the efficiency of deep parsers with incremental chart pruning. In *Proceedings of the ESSLLI Workshop on Parsing with Categorial Grammars*, Bordeaux, France, 2009. (in press).
7. P. Lison and G.-J. M. Kruijff. Salience-driven contextual priming of speech recognition for human-robot interaction. In *Proceedings of the 18th European Conference on Artificial Intelligence*, Patras (Greece), 2008.
8. M. Steedman and J. Baldridge. Combinatory categorial grammar. In Robert Borsley and Kersti Börjars, editors, *Nontransformational Syntax: A Guide to Current Models*. Blackwell, Oxford, 2009.
9. K. Weilhammer, M. N. Stuttle, and S. Young. Bootstrapping language models for dialogue systems. In *Proceedings of INTERSPEECH 2006*, Pittsburgh, PA, 2006.
10. L. S. Zettlemoyer and M. Collins. Online learning of relaxed CCG grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 678–687, 2007.