Graph-to-Graph Meaning Representation Transformations for Human-Robot Dialogue

Anonymous NAACL submission

Introduction. In our research in support of two-way human-robot communication in the search and navigation domain, we leverage Abstract Meaning Representation (AMR) to capture and structure the semantic content of natural language instructions in a machine-readable, directed, a-cyclic graph (Banarescu et al., 2013). To effectively map AMR to a constrained robot action specification while maintaining all necessary elements for understanding, we developed a set of in-domain, task-specific AMR graphs augmented with speech act and tense and aspect information not found in the original AMR. Here, we detail our efforts to exploit both rule-based and classifier-based methods to transform AMR graphs into the in-domain graphs, thereby bridging the gap from entirely unconstrained natural language input to a fixed set of robot actions (see Fig. 1).

Background: Data & Annotations. We used a corpus of human-robot dialogue in which a person directs a remotely located robot to complete search and navigation tasks (Marge et al., 2016). We manually selected 504 utterances made up of short, sequential excerpts of the corpus data representative of the variety of common exchange types that we see. These sentences were independently double-annotated (IAA 87.8% using the Smatch metric (Cai and Knight, 2013)) and adjudicated following current AMR guidelines.1

For a subset of this data (roughly 300 utterances), one annotator manually converted the AMR graph into one of 36 in-domain templates specific to this task. These “templates” are skeletal AMRs in which the top, anchor node is a fixed relation corresponding to a speech act type (e.g., assert-02 in the AMR lexicon); one of its numbered arguments, or ‘ARGs,’ is a fixed relation corresponding to an action (e.g., turn-01) or the content of the speech act; and arguments of these relations are filled out to detail both dialogue relationships (utterance level) and action specifications (content level).2 These 36 speech acts are classified into 5 types, listed in Fig. 2 (number of subtypes in parentheses), along with example subtypes for the type command.

Graph-to-Graph Transformations. We convert parser output AMRs into our in-domain templates through a mixed methods approach: rule-based and classifier-based systems. While there exists a neural AMR graph converter for a related task (Liu et al., 2015), neural systems require substantial training data. Alternatively, our approach leverages the highly structured information in the AMR graph and a relatively small data set.

Speech Act Classifier. An initial step in the graph-to-graph transformation requires a speech act classifier to predict one of the five speech act types from an utterance and thereby trigger the appropriate in-domain template for use. Since natural language is variable, we implement a classifier that will be robust to any language input, rather than rely on a rule-based approach in this step. The word move, for instance, can signal a command (Robot, move forward...), but it can also be present in assertions (I moved three feet) and questions (Is it possible to move around that cone?). We implemented an off-the-shelf Naive Bayes multinomial classifier for our baseline from the scikit-learn library, using bigrams as features (Pedregosa et al.).

Rule-Based Slot Filling. The rule-based aspect of this pipeline relies on regular expressions to find and extract portions of the original AMR to fill the appropriate slots in the in-domain template. For example, in Fig. 1, the command:move template is triggered with the relation command-02 anchoring the template and the relation go-02 capturing the impelled action, ARG2, of command-02. In our restricted domain, the ARG0 agent slot of command-02 is fixed as the Commander, and ARG1 entity-commanded as the Robot. What is treated as both ARG0 (mover) and ARG1 (moved) for move-01 in the original graph (here, you) is converted into the ARG0 self-directed mover of go-02, and this slot is reassigned to Robot in the domain representation. The system then looks for the required end point ARG4 slot in the original AMR. That slot may be realized as a leaf concept node, door in this case, or a subgraph containing multiple slots for more descriptive endpoints. The precise rules vary depending upon the template triggered, as well as

1https://github.com/amrisi/amr-guidelines/blob/master/amr.md
2Full details published elsewhere.
Figure 1: Pipeline—Instructions are parsed into AMR using parsers, then converted via graph-to-graph transformation into one of our augmented AMR templates and, if executable, mapped to a robot action specifications.

**SPEECH ACT TYPES**

| c   | command (6) | command:move |
| a   | assert (9)  | command:turn |
| r   | request (4) | command:send-image |
| q   | question (3) | command:repeat |
| e   | express (5) | command:cancel |
|     |             | command:stop |

Figure 2: Speech act types with example subtypes.

the original relation used (e.g., move, drive, proceed), which all merge to a consistent predicate in the template (e.g., go-02).

**Results.** Speech act classifier performance using 10-fold cross-validation is presented in Table 1. Overall accuracy for the speech act classifier is relatively high, which is expected, given the restricted domain and highly repetitive phrases. **Question** and **Request** were the most challenging categories due to higher variation of language and some overlap in the language of the two categories and commands. For example, **Can you describe it another way?** could be seen as a polite command, a request, or a question even to human annotators; thus, we are also evaluating the quality of the speech act distinctions.

| Speech Act | Precision | Recall | F-1 |
|------------|-----------|--------|-----|
| Assert     | .96       | .96    | .96 |
| Command    | .98       | .94    | .96 |
| Question   | .69       | .81    | .71 |
| Request    | .70       | .92    | .76 |
| Express    | .94       | .83    | .86 |

**Table 1: Speech act classifier performance**

Separately, we evaluated the rule-based transformations by comparing output graphs to the manually annotated gold standard for a sample of the most prevalent commands (move, turn, and send-image subtypes). Smatch scores (reported in Table 2) are also reasonably high.\(^3\) Performance for **command:move** is the worst, due to the vari-

| Command Subtype | Number | Smatch |
|-----------------|--------|--------|
| Move            | 16     | 78.5   |
| Turn            | 20     | 91.5   |
| Send-Image      | 13     | 89.2   |

**Table 2: Rule-based transformation Smatch evaluation.**

**Conclusions & Future Work.** The preliminary results presented here are quite promising, as reflected by high F-1 and Smatch scores. However, we have yet to see how this translates into performance in the end-to-end system we are now working to implement. Remaining challenges include handling truly ambiguous speech acts that cannot be determined from the language alone, which we hope to resolve by leveraging dialogue context and computer vision in the future.

**References**

L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider. 2013. Abstract Meaning Representation for sembanking. In Proc. LAW, pages 178–186.

S. Cai and K. Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In Proc. of ACL, volume 2, pages 748–752.

F. Liu, J. Flanigan, S. Thomson, N. Sadeh, and N. A. Smith. 2015. Toward abstractive summarization using semantic representations. In Proc. of NAACL-HLT.

M. Marge, C. Bonial, B. Byrne, T. Cassidy, W. Evans, S. Hill, and C. Voss. 2016. Applying the wizard-of-oz technique to multimodal human-robot dialogue. In Proc. of RO-MAN.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, and O. Grisel et al. Scikit-learn: Machine Learning in Python.\(^3\)IAA Smatch scores are generally 70-80%, AMR development group communication, 2014.