AT&T: The Tag&Parse Approach to Semantic Parsing of Robot Spatial Commands

Svetlana Stoyanchev, Hyuckchul Jung, John Chen, Srinivas Bangalore
AT&T Labs Research
1 AT&T Way Bedminster NJ 07921
{sveta,hjung,jchen,srini}@research.att.com

Abstract

The *Tag&Parse* approach to semantic parsing first assigns semantic tags to each word in a sentence and then parses the tag sequence into a semantic tree. We use statistical approach for tagging, parsing, and reference resolution stages. Each stage produces multiple hypotheses which are re-ranked using spatial validation. We evaluate the *Tag&Parse* approach on a corpus of Robotic Spatial Commands as part of the SemEval Task6 exercise. Our system accuracy is 87.35% and 60.84% with and without spatial validation.

1 Introduction

In this paper we describe a system participating in the SemEval2014 Task-6 on Supervised Semantic Parsing of Robotic Spatial Commands. It produces a semantic parse of natural language commands addressed to a robot arm designed to move objects on a grid surface. Each command directs a robot to change position of an object given a current configuration. A command uniquely identifies an object and its destination, for example *"Move the turquoise pyramid above the yellow cube"*. System output is a Robot Control Language (RCL) parse (see Figure 1) which is processed by the robot arm simulator. The Robot Spatial Commands dataset (Dukes, 2013) is used for training and testing.

Our system uses a *Tag&Parse* approach which separates semantic tagging and semantic parsing stages. It has four components: 1) semantic tagging, 2) parsing, 3) reference resolution, and 4) spatial validation. The first three are trained using LLAMA (Haffner, 2006), a supervised machine learning toolkit, on the RCL-parsed sentences.

For semantic tagging, we train a maximum entropy sequence tagger for assigning a semantic label and value to each word in a sentence, such as *type_cube* or *color_blue*. For parsing, we train a constituency parser on non-lexical RCL semantic trees. For reference resolution, we train a maximum entropy model that identifies entities for *reference* tags found by previous components. All of these components can generate multiple hypotheses. Spatial validation re-ranks these hypotheses by validating them against the input spatial configuration. The top hypothesis after re-ranking is returned by the system.

Separating tagging and parsing stages has several advantages. A tagging stage allows the system flexibility to abstract from possible grammatical or spelling errors in a command. It assigns a semantic category to each word in a sentence. Words not contributing to the semantic meaning are assigned 'O' label by the tagger and are ignored in the further processing. Words that are misspelled can potentially receive a correct tag when a word similarity feature is used in building a tagging model. This will be especially important when processing output of spoken commands that may contain recognition errors.

The remainder of the paper is organized thusly. In Section 2 we describe each of the components used in our system. In Section 3 we describe the results reported for SemEval2014 and evaluation of each system component. We summarize our findings and present future work in Section 4.

2 System

2.1 Sequence Tagging

A sequence tagging approach is used for conditional inference of tags given a word sequence. It is used for many natural language tasks, such as part of speech (POS) and named entity tagging (Toutanova and others, 2003; Carreras et al., 2003). We train a sequence tagger for assign-

Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 109–113,
Dublin, Ireland, August 23-24, 2014.

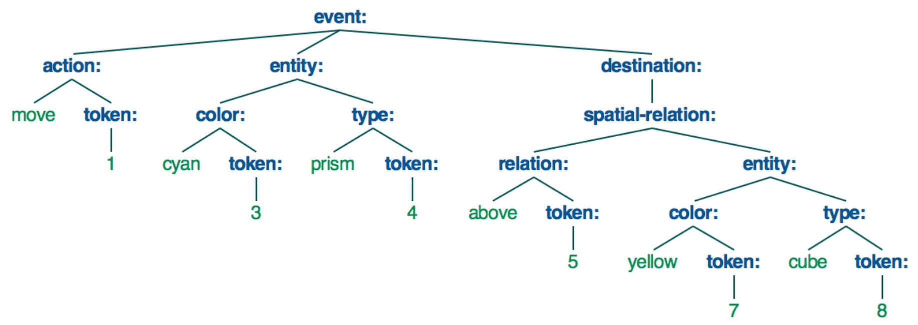

Figure 1: RCL tree for a sentence *Move the turquoise pyramid above the yellow cube.*

Word	index	tag	label
Move	1	action	move
the	2	O	-
turquoise	3	color	cyan
pyramid	4	type	prism
above	5	relation	above
the	6	O	-
yellow	7	color	yellow
cube	8	type	cube

Table 1: Tagging labels for a sentence *Move the turquoise pyramid above the yellow cube.*

ing a combined semantic tag and label (such as *type_cube*) to each word in a command. The tags used for training are extracted from the leaf-level nodes of the RCL trees. Table 2 shows tags and labels for a sample sentence *"Move the turquoise pyramid above the yellow cube"* extracted from the RCL parse tree (see Figure 1). In some cases, a label is the same as a word (yellow, cube) while in other cases, it differs (turquoise - cyan, pyramid - prism).

We train a sequence tagger using LLAMA maximum entropy (maxent) classification (Haffner, 2006) to predict the combined semantic tag and label of each word. Neighboring words, immediately neighboring semantic tags, and POS tags are used as features, where the POS tagger is another sequence tagging model trained on the Penn Treebank (Marcus et al., 1993). We also experimented with a tagger that assigns tags and labels in separate sequence tagging models, but it performed poorly.

2.2 Parsing

We use a constituency parser for building RCL trees. The input to the parser is a sequence of tags assigned by a sequence tagger, such as *"action color type relation color type"* for the example in Figure 1.

The parser generates multiple RCL parse tree hypotheses sorted in the order of their likelihood. The likelihood of a tree T given a sequence of tags T is determined using a probabilistic context free grammar (PCFG) G:

$$P(T|S) = \prod_{r \in T} P_G(r) \qquad (1)$$

The n-best parses are obtained using the CKY algorithm, recording the n-best hyperedge backpointers per constituent along the lines of (Huang and Chiang, 2005). G was obtained and P_G was estimated from a corpus of non-lexical RCL trees generated by removing all nodes descendant from the tag nodes (action, color, etc.). Parses may contain empty nodes not corresponding to any tag in the input sequence. These are hypothesized by the parser at positions in between input tags and inserted as edges according to the PCFG, which has probabilistic rules for generating empty nodes.

2.3 Reference Resolution

Reference resolution identifies the most probable antecedent for each anaphor within a text (Hirschman and Chinchor, 1997). It applies when multiple candidates antecedents are present. For example, in a sentence *"Pick up the red cube standing on a grey cube and place it on top of the yellow one"*, the anaphor *it* has two candidate antecedents corresponding to entity segments *the red cube* and *a grey cube*. In our system, anaphor and antecedents are represented by *reference tags* occurring in one sentence. A reference tag is either assigned by a sequence tagger to one of the words (e.g. to a pronoun) or is inserted into a tree by the parser (e.g. ellipsis). We train a binary maxent model for this task using LLAMA. The input is a pair consisting of an anaphor and a candidate antecedent, along with their features.

Features that are used include the preceding and following words as well as the tags/labels of both the anaphor and candidate antecedent. The reference resolution component selects the antecedent for which the model returns the highest score.

2.4 Spatial Validation

SemEval2014 Task6 provided a spatial planner which takes an RCL command as an input and determines if that command is executable in the given spatial context. At each step described in 2.1∼2.3, due to the statistical nature of our approach, multiple hypotheses can be easily computed with different confidence values. We used the spatial planner to validate the final output RCL commands from the three steps by checking if the RCLs are executable or not. We generate multiple tagger output hypotheses. For each tagger output hypothesis, we generate multiple parser output hypotheses. For each parser output hypothesis, we generate multiple reference resolution output hypotheses. The resulting output hypotheses are ranked in the order of confidence scores with the highest tagging output scores ranked first, followed by the parsing output scores, and, finally, reference resolution output scores. The system returns the result of the top scored command that is valid according to the spatial validator.

In many applications, there can be a tool or method to validate tag/parse/reference outputs fully or partially. Note that in our system the validation is performed after all output is generated. Tightly coupled validation, such as checking validity of a tagged entity or a parse constituent, could help in computing hypotheses at each step (e.g., feature values based on possible entities or actions) and it remains as future work.

3 Results

In this section, we present evaluation results on the three subsets of the data summarized in Table 3. In the TEST2500 data set, the models are trained on the initial 2500 sentences of the Robot Commands Treebank and evaluated on the last 909 sentences (this corresponds to the data split of the SemEval task). In TEST500 data set, the models are trained on the initial 500 sentences of the training set and evaluated on the last 909 test sentences. We report these results to analyze the models' performance on a reduced training size. In DEV2500 data set, models are trained on 90% of the initial 2500 sentences and evaluated on 10% of the 2500 sentences using a random data split. We observe that sentence length and standard deviation of test sentences in the TEST2500 data set is higher than on the training sentences while in the DEV2500 data set training and test sentence length and standard deviation are comparable.

#	Dataset	Avg # hyp	Accuracy
1	TEST2500 1-best	1	86.0%
2	TEST2500 max-5	3.34	95.2%
3	TEST500 1-best	1	67.9%
4	TEST500 max-5	4.25	83.8%
5	DEV2500 1-best	1	90.8%
6	DEV2500 max-5	2.9	98.0%

Table 3: Tagger accuracy for 1-best and maximum of 5-best hypotheses (max-5).

3.1 Semantic Tagging

Table 3 presents sentence accuracy of the semantic tagging stage. Tagging accuracy is evaluated on 1-best and on max-5 best tagger outputs. In the max-5 setting the number of hypotheses generated by the tagger varies for each input with the average numbers reported in Table 3. Tagging accuracy on TEST2500 using 1-best is 86.0%. Considering max-5 best tagging sequences, the accuracy is 95.2%. On the TEST500 data set tagging accuracy is 67.9% and 83.8% on 1-best and max-5 best sequences respectively, approximately 8% points lower than on TEST2500 data set. On the DEV2500 data set tagging accuracy is 90.8% and 98.0% on 1-best and max-5 best sequences, 4.8% and 2.8% points higher than on the TEST2500 data set. The higher performance on DEV2500 in comparison to the TEST2500 can be explained by the higher complexity of the test sentences in comparison to the training sentences in the TEST2500 data set.

3.2 RCL Parsing

Parsing was evaluated using the EVALB scoring metric (Collins, 1997). Its 1-best F-measure accuracy on gold standard TEST2500 and DEV2500 semantic tag sequences was 96.17% and 95.20%, respectively. On TEST500, its accuracy remained 95.20%. On TEST2500 with system provided input sequences, its accuracy was 94.79% for 869 out of 909 sentences that were tagged correctly.

3.3 System Accuracy

Table 4 presents string accuracy of automatically generated RCL parse trees on each data set. The

Name	Train #sent	Train Sent. len. (stdev)	Test #sent	Test Sent. Len. (stdev)
TEST2500	2500	13.44 (5.50)	909	13.96 (5.59)
TEST500	500	14.62(5.66)	909	13.96 (5.59)
DEV2500	2250	13.43 (5.53)	250	13.57 (5.27)

Table 2: Number of sentences, average length and standard deviation of the data sets.

results are obtained by comparing system output RCL parse string with the reference RCL parse string. For each data set, we ran the system with and without spatial validation. We ran RCL parser and reference resolution on automatically assigned semantic tags (Auto) and oracle tagging (Orcl). We observed that some tag labels can be verified systematically and corrected them with simple rules: e.g., change "front" to "forward" because relation specification in (Dukes, 2013) doesn't have "front" even though annotations included cases with "front" as relation.

The system performance on TEST2500 data set using automatically assigned tags and no spatial validation is 60.84%. In this mode, the system uses 1-best parser and 1-best tagger output. With spatial validation, which allows the system to re-rank parser and tagger hypotheses, the performance increases by 27% points to 87.35%. This indicates that the parser and the tagger component often produce a correct output which is not ranked first. Using oracle tags without / with spatial validation on TEST2500 data set the system accuracy is 67.55% / 94.83%, 7% points above the accuracy using predicted tags.

The system performance on TEST500 data set using automatically assigned tags with / without spatial validation is 48.95% / 74.92%, approximately 12% points below the performance on TEST2500 (Row 1). Using oracle tags without / with spatial validation the performance on TEST500 data set is 63.89% / 94.94%. The performance without spatial validation is only 4% below TEST2500, while with spatial validation the performance on TEST2500 and TEST500 is the same. These results indicate that most performance degradation on a smaller data set is due to the semantic tagger.

The system performance on DEV2500 data set using automatically assigned tags without / with spatial validation is 68.0% / 96.80% (Row 5), 8% points above the performance on TEST2500 (Row 1). With oracle tags, the performance is 69.60% / 98.0%, which is 2-3% points above TEST2500 (Row 2). These results indicate that most performance improvement on a better balanced data set DEV2500 is due to better semantic tagging.

#	Dataset	Tag	Accuracy without / with spatial validation
1	TEST2500	Auto	60.84 / 87.35
2	TEST2500	Orcl	67.55 / 94.83
3	TEST500	Auto	48.95 / 74.92
4	TEST500	Orcl	63.89 / 94.94
5	DEV2500	Auto	68.00 / 96.80
6	DEV2500	Orcl	69.60 / 98.00

Table 4: System accuracy with and without spatial validation using automatically assigned tags and oracle tags (OT).

4 Summary and Future Work

In this paper, we present the results of semantic processing for natural language robot commands using *Tag&Parse* approach. The system first tags the input sentence and then applies non-lexical parsing to the tag sequence. Reference resolution is applied to the resulting parse trees. We compare the results of the models trained on the data sets of size 500 (TEST500) and 2500 (TEST2500) sentences. We observe that sequence tagging model degrades significantly on a smaller data set. Parsing and reference resolution models, on the other hand, perform nearly as well on both training sizes. We compare the results of the models trained on more (DEV2500) and less (TEST2500) homogeneous training/testing data sets. We observe that a semantic tagging model is more sensitive to the difference between training and test set than parsing model degrading significantly a less homogeneous data set. Our results show that 1) both tagging and parsing models will benefit from an improved re-ranking, and 2) our parsing model is robust to a data size reduction while tagging model requires a larger training data set.

In future work we plan to explore how *Tag&Parse* approach will generalize in other domains. In particular, we are interested in using a combination of domain-specific tagging models and generic semantic parsing (Das et al., 2010) for processing spoken commands in a dialogue system.

References

Xavier Carreras, Lluís Màrquez, and Lluís Padró. 2003. A Simple Named Entity Extractor Using AdaBoost. In *Proceedings of the CoNLL*, pages 152–157, Edmonton, Canada.

Michael Collins. 1997. Three Generative Lexicalized Models for Statistical Parsing. In *Proceedings of the 35th Annual Meeting of the ACL*, pages 16–23.

Dipanjan Das, Nathan Schneider, Desai Chen, and Noah A. Smith. 2010. Probabilistic Frame-Semantic Parsing. In *HLT-NAACL*, pages 948–956.

Kais Dukes. 2013. Semantic Annotation of Robotic Spatial Commands. In *Language and Technology Conference (LTC)*.

Patrick Haffner. 2006. Scaling large margin classifiers for spoken language understanding. *Speech Communication*, 48(3-4):239–261.

Lynette Hirschman and Nancy Chinchor. 1997. MUC-7 Coreference Task Definition. In *Proceedings of the Message Understanding Conference (MUC-7)*. Science Applications International Corporation.

Liang Huang and David Chiang. 2005. Better K-best Parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology*, Parsing '05, pages 53–64, Stroudsburg, PA, USA.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of the 2003 Conference of the NAACL on Human Language Technology - Volume 1*, pages 173–180.