SCHWA: PETE using CCG Dependencies with the C&C Parser

Dominick Ng, James W. D. Constable, Matthew Honnibal and **James R. Curran**
ə-lab, School of Information Technologies
University of Sydney
NSW 2006, Australia
{dong7223,jcon6353,mhonn,james}@it.usyd.edu.au

Abstract

This paper describes the SCHWA system entered by the University of Sydney in SemEval 2010 Task 12 – Parser Evaluation using Textual Entailments (Yuret et al., 2010). Our system achieved an overall accuracy of 70% in the task evaluation.

We used the C&C parser to build CCG dependency parses of the truth and hypothesis sentences. We then used partial match heuristics to determine whether the system should predict entailment. Heuristics were used because the dependencies generated by the parser are construction specific, making full compatibility unlikely. We also manually annotated the development set with CCG analyses, establishing an upper bound for our entailment system of 87%.

1 Introduction

The SemEval 2010 Parser Evaluation using Textual Entailments (PETE) task attempts to address the long-standing problems in parser evaluation caused by the diversity of syntactic formalisms and analyses in use. The task investigates the feasibility of a minimalist extrinsic evaluation – that of detecting textual entailment between a truth sentence and a hypothesis sentence. It is extrinsic in the sense that it evaluates parsers on a task, rather than a direct comparison of their output against some gold standard. However, it requires only minimal task-specific logic, and the proposed entailments are designed to be inferrable based on syntactic information alone.

Our system used the C&C parser (Clark and Curran, 2007a), which uses the Combinatory Categorial Grammar formalism (CCG, Steedman, 2000). We used the CCGbank-style dependency output of the parser (Hockenmaier and Steedman, 2007), which is a directed graph of head-child relations labelled with the head's lexical category and the argument slot filled by the child.

We divided the dependency graphs of the truth and hypothesis sentences into *predicates* that consisted of a head word and its immediate children. For instance, the parser's analysis of the sentence *Totals include only vehicle sales reported in period* might produce predicates like *include(Totals, sales)*, *only(include)*, and *reported(sales)*. If at least one such predicate matches in the two parses, we predict entailment. We consider a single predicate match sufficient for entailment because the lexical categories and slots that constitute our dependency labels are often different in the hypothesis sentence due to the generation process used in the task.

The single predicate heuristic gives us an overall accuracy of 70% on the test set. Our precision and recall over the test set was 68% and 80% respectively giving an F-score of 74%.

To investigate how many of the errors were due to parse failures, and how many were failures of our entailment recognition process, we manually annotated the 66 development truth sentences with gold standard CCG derivations. This established an upper bound of 87% F-score for our approach.

This upper bound suggests that there is still work to be done before the system allows transparent evaluation of the parser. However, cross-framework parser evaluation is a difficult problem: previous attempts to evaluate the C&C parser on grammatical relations (Clark and Curran, 2007b) and Penn Treebank-trees (Clark and Curran, 2009) have also produced upper bounds between 80 and 90% F-score. Our PETE system was much easier to produce than either of these previous attempts at cross-framework parser evaluation, suggesting that this may be a promising approach to a difficult problem.

Proceedings of the 5th International Workshop on Semantic Evaluation, ACL 2010, pages 313–316,
Uppsala, Sweden, 15-16 July 2010.

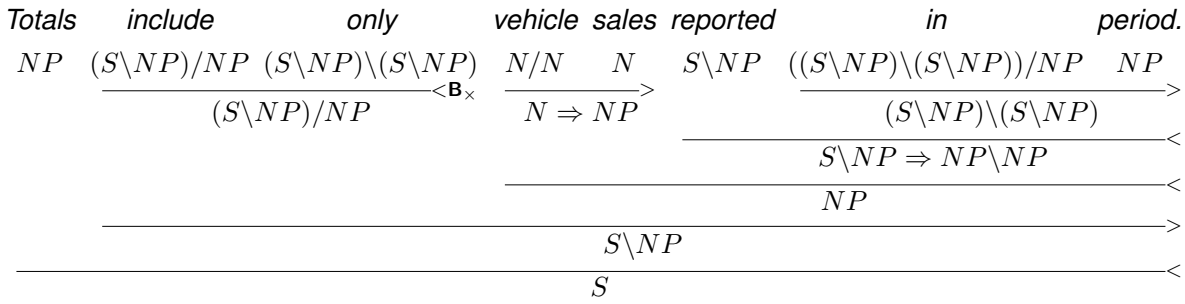

Figure 1: An example CCG derivation, showing how the categories assigned to words are combined to form a sentence. The arrows indicate the direction of application.

2 Background

Combinatory Categorial Grammar (CCG, Steedman, 2000) is a lexicalised grammar formalism based on combinatory logic. The grammar is directly encoded in the lexicon in the form of combinatory categories that govern how each word combines with its neighbours. The parsing process determines the most likely assignment of categories to words, and finds a sequence of combinators that allows them to form a sentence.

A sample CCG derivation for a sentence from the test set is shown in Figure 1. The category for each word is indicated beneath it. It can be seen that some categories take other categories as arguments; each argument slot in a category is numbered based on the order of application, from latest to earliest. For example:

$$((S/NP_1)/(S/NP)_2)\backslash NP_3$$

Figure 2 shows how the argument slots are mapped to dependencies. The first two columns list the predicate words and their categories, while the second two show how each argument slot is filled. For example, in the first row, *only* has the category $(S\backslash NP)\backslash(S\backslash NP)$, with argument slot 1 filled by *include*). It is these dependencies that form the basis for our predicates in this task.

only	$(S\backslash NP)\backslash(S\backslash NP)$	1	*include*
vehicle	N/N	1	*sales*
in	$((S\backslash NP)\backslash(S\backslash NP))/NP$	2	*period*
in	$((S\backslash NP)\backslash(S\backslash NP))/NP$	1	*reported*
reported	$S\backslash NP$	1	*sales*
include	$(S\backslash NP)/NP$	2	*sales*
include	$(S\backslash NP)/NP$	1	*Totals*

Figure 2: The dependencies represented by the derivation in Figure 1.

Recent work has seen the development of high-performance parsers built on the CCG formalism. Clark and Curran (2007a) demonstrate the use of techniques like adaptive supertagging, parallelisation and a dynamic-programming chart parsing algorithm to implement the C&C parser, a highly efficient CCG parser that performs well against parsers built on different formalisms (Rimell et al., 2009). We use this parser for the PETE task.

The performance of statistical parsers is largely a function of the quality of the corpora they are trained on. For this task, we used models derived from the CCGbank corpus – a transformation of the Penn Treebank (Marcus et al., 1993) including CCG derivations and dependencies (Hockenmaier, 2003a). It was created to further CCG research by providing a large corpus of appropriately annotated data, and has been shown to be suitable for the training of high-performance parsers (Hockenmaier, 2003b; Clark and Curran, 2004).

3 Method

Our system used the C&C parser to parse the truth and hypothesis sentences. We took the dependencies generated by the parser and processed these to generate predicates encoding the canonical form of the head word, its required arguments, and their order. We then attempted to unify the predicates from the hypothesis sentence with the predicates in the truth sentence. A successful unification of predicates a and b occurs when the head words of a and b are identical and their argument slots are also identical. If any predicate from the hypothesis sentence unified with a predicate from the truth sentence, our system returned YES, otherwise the system returned NO.

We used the 66 sentence development set to tune our approach. While analysing the hypothesis sentences, we noticed that many examples re-

System	YES entailment correct	incorrect	A (%)	NO entailment correct	incorrect	A (%)	Overall accuracy (%)	F-score
SCHWA	125	31	80	87	58	60	70	74
median	71	85	46	88	57	61	53	50
baseline	156	0	100	0	145	0	52	68
low	68	88	44	76	69	52	48	46

Table 1: Final results over the test set

System	YES entailment correct	incorrect	A (%)	NO entailment correct	incorrect	A (%)	Overall accuracy (%)	F-score
Gold deps	34	6	85	22	4	90	87	87
Parsed deps	32	8	80	20	6	77	79	82

Table 2: Results over the development set

placed nouns from the truth sentence with indefinite pronouns such as *someone* or *something* (e.g. *Someone bought something*). In most of these cases the indefinite would not be present in the truth sentence at all, so to deal with this we converted indefinite pronouns into wildcard markers that could be matched to any argument. We also incorporated sensitivity to passive sentences by adjusting the argument numbers of dependents.

In its most naive form our system is heavily biased towards excellent recall but poor precision. We evaluated a number of heuristics to prune the predicate space and selected those which improved the performance over the development set. Our final system used the part-of-speech tags generated by the parser to remove predicates headed by determiners, prepositions and adjectives. We note that even after predicate pruning our system is still likely to return better recall performance than precision, but this discrepancy was masked in part by the nature of the development set: most hypotheses are short and so the potential number of predicates after pruning is likely to be small. The final predicates generated by the system for the example derivation given in Figure 1 after heuristic pruning are:

```
only(include)
reported(sales)
include(totals, sales)
```

4 Results

We report results over the 301 sentence test set in Table 1. Our overall accuracy was 70%, and performance over YES entailments was roughly 20% higher than accuracy over NO entailments. This bias towards YES entailments is a reflection of our single match heuristic that only required one predicate match before answering YES. Our system performed nearly 20% better than the baseline system (all YES responses) and placed second overall in the task evaluation.

Table 2 shows our results over the development corpus. The 17% drop in accuracy and 8% drop in F-score between the development data and the test data suggests that our heuristics may have overfitted to the limited development data. More sophisticated heuristics over a larger corpus would be useful for further fine-tuning our system.

4.1 Results with Gold Standard Parses

Our entailment system's errors could be broadly divided into two classes: those due to incorrect parses, and those due to incorrect comparison of the parses. To investigate the relative contributions of these two classes of errors, we manually annotated the 66 development sentences with CCG derivations. This allowed us to evaluate our system using gold standard parses. Only one annotator was available, so we were unable to calculate inter-annotator agreement scores to examine the quality of our annotations.

The annotation was prepared with the annotation tool used by Honnibal et al. (2009). The tool presents the user with a CCG derivation produced by the C&C parser. The user can then correct the lexical categories, or add bracket constraints to the parser using the algorithm described by Djordjevic and Curran (2006), and reparse the sentence until the derivation desired is produced.

Our results with gold standard dependencies are

shown in Table 2. The accuracy is 87%, establishing a fairly low upper bound for our approach to the task. Manual inspection of the remaining errors showed that some were due to incorrect parses for the hypothesis sentence, and some were due to entailments which the parser's dependency analyses could not resolve, such as *They ate whole steamed grains ⇒ The grains were steamed*. The largest source of errors was our matching heuristics, suggesting that our approach to the task must be improved before it can be considered a transparent evaluation of the parser.

5 Conclusion

We constructed a system to evaluate the C&C parser using textual entailments. We converted the parser output into a set of predicate structures and used these to establish the presence of entailment. Our system achieved an overall accuracy of 79% on the development set and 70% over the test set. The gap between our development and test accuracies suggests our heuristics may have been overfitted to the development data.

Our investigation using gold-standard dependencies established an upper bound of 87% on the development set for our approach to the task. While this is not ideal, we note that previous efforts at cross-parser evaluation have shown that it is a difficult problem (Clark and Curran (2007b) and Clark and Curran (2009)). We conclue that the concept of a minimal extrinsic evaluation put forward in this task is a promising avenue for formalism-independent parser comparison.

References

Stephen Clark and James R. Curran. Parsing the WSJ using CCG and log-linear models. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 104–111, 2004.

Stephen Clark and James R. Curran. Wide-Coverage Efficient Statistical Parsing with CCG and Log-Linear Models. *Computational Linguistics*, 33(4):493–552, 2007a.

Stephen Clark and James R. Curran. Formalism-independent parser evaluation with CCG and DepBank. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 248–255, Prague, Czech Republic, 25–27 June 2007b.

Stephen Clark and James R. Curran. Comparing the accuracy of CCG and Penn Treebank Parsers. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 53–56, Suntec, Singapore, August 2009.

Bojan Djordjevic and James R. Curran. Faster wide-coverage CCG parsing. In *Proceedings of the Australasian Language Technology Workshop 2006*, pages 3–10, Sydney, Australia, December 2006.

Julia Hockenmaier. *Data and models for statistical parsing with Combinatory Categorial Grammar*. PhD thesis, 2003a.

Julia Hockenmaier. Parsing with generative models of predicate-argument structure. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 359–366. Association for Computational Linguistics Morristown, NJ, USA, 2003b.

Julia Hockenmaier and Mark Steedman. CCGbank: a corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396, 2007.

Matthew Honnibal, Joel Nothman, and James R. Curran. Evaluating a Statistical CCG Parser on Wikipedia. In *Proceedings of the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources*, pages 38–41, Singapore, August 2009.

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

Laura Rimell, Stephen Clark, and Mark Steedman. Unbounded Dependency Recovery for Parser Evaluation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, volume 2, pages 813–821, 2009.

Mark Steedman. *The Syntactic Process*. MIT Press, Massachusetts Institute of Technology, USA, 2000.

Deniz Yuret, Aydın Han, and Zehra Turgut. SemEval-2010 Task 12: Parser Evaluation using Textual Entailments. In *Proceedings of the SemEval-2010 Evaluation Exercises on Semantic Evaluation*, 2010.