Integrating Verb-Particle Constructions into CCG Parsing

James W. D. Constable and **James R. Curran**
School of Information Technologies
University of Sydney
NSW 2006, Australia
{jcon6353,james}@it.usyd.edu.au

Abstract

Despite their prevalence in the English language, multiword expressions like verb-particle constructions (VPCs) are often poorly handled by NLP systems. This problem is partly due to inadequacies in existing corpora; the primary corpus for CCG-oriented work, CCGbank, does not account for VPCs at all, and is inconsistent in its handling of them. In this paper, we apply some corrective transformations to CCGbank, and then use it to retrain an augmented version of the Clark and Curran CCG parser. Using our technique, we observe no significant change in F-score, while the resulting parse is semantically more sound.

1 Introduction

Multiword expressions (MWEs), compound lexemes made up of two or more words that together form a complete semantic unit, are one of the problems facing natural language processing systems. Verb-particle constructions (VPCs) are a common type of MWE, comprising a verb and a particle, most often a preposition. The meaning of some VPCs can be logically attributed to the component parts (e.g., *picked out*), but many are idiomatic and semantically opaque (e.g., *make out*).

Previous research into VPCs has focussed much attention on their automatic extraction and classification (Baldwin and Villavicencio, 2002; Villavicencio, 2003). However, research into how they should be handled by parsers is noticeably lacking. Their unusual ability to manifest in both a 'joined' and 'split' configuration ('*gunned down the man*' versus '*gunned the man down*') prevents parsers from treating them as a single unit, and demands a system that is able to maintain the semantic bond between the components, even when they are non-adjacent.

To compound the problem, existing corpora are not consistent in their handling of these constructions. The Penn Treebank (Marcus et al., 1993, 1994) has an RP tag for particles, but sometimes labels them as adverbs. The CCGbank (Hockenmaier and Steedman, 2007) analysis of particles varies, but leans towards treating all particles as adverbial modifiers. This is in itself problematic, since it fails to take into account the fact that particles are a core part of the construct, whereas adverbs are optional. This lack of quality corpora for VPC-related work limits the power of corpus-trained parsers.

In this paper we draw on the Penn Treebank and PropBank (Kingsbury and Palmer, 2003) to repair CCGbank's representation of VPCs, and demonstrate how our approach is able to satisfactorily account for most VPC-related phenomena. Retraining the Clark and Curran parser (Clark and Curran, 2007) on our modified corpus, we observe a very slight decrease in parser F-score, although this is balanced by the fact that the parses now make structural sense.

2 Combinatory Categorial Grammar

Combinatory Categorial Grammar (CCG, Steedman (2000)) is a lexicalised grammar formalism based on combinatory logic. One of the features that makes CCG so appealing to NLP researchers is its high degree of *lexicalisation* (i.e., the degree to which the grammar is built into the lexicon). Every word is assigned a category, and parsing is simply a matter of finding the right sequence of combinators to form a sentence. Recent work has seen the creation of high-performance parsers built on the CCG formalism (Clark and Curran, 2007).

The primary corpus for CCG-related work is CCGbank — an augmented version of the Penn Treebank (Marcus et al., 1993) that contains CCG derivations and predicate argument structures. It was induced from the Penn Treebank (Hocken-

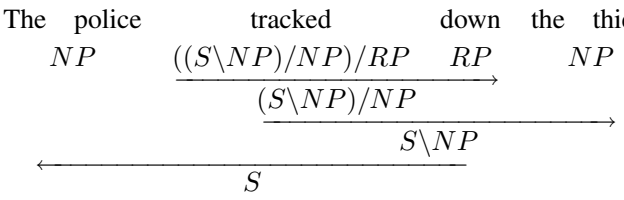

Figure 1: The default case — a VPC in the joined configuration.

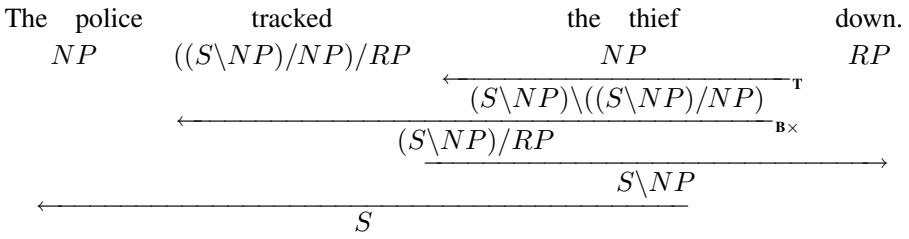

Figure 2: Using type-raising and backward-crossed-composition to handle the split configuration.

maier, 2003) with the goal of furthering CCG research by providing a large corpus of suitably annotated data.

Despite its utility, CCGbank is not without its flaws. Hockenmaier explains that the Treebank does not contain enough information to perform a perfect automatic translation to CCG, and points to complement/adjunct distinctions, phrasal verbs and compound nouns as problematic areas. Some attempts have been made to rectify this; for example, Honnibal and Curran (2007) target the complement/adjunct distinction. Most relevant to our work is the failure to capture phrasal verbs, resulting in the unfortunate situation of particles being treated as adverbial modifiers, and verbs failing to subcategorise for them.

3 CCG Representation of VPCs

Before modifying CCGbank, we first had to determine a suitable method of representing VPCs in CCG. The representation would ideally minimise the ambiguity of the lexical categories, and maintain CCG's transparent interface between syntax and semantics.

The current representation in CCGbank tends to favour an adverb-style treatment, where the verb is assigned a normal verbal category, and the particle is given the category $(S\backslash NP)\backslash(S\backslash NP)$ (i.e. a post-modifying adverb). This approach is semantically rather unsatisfying. The particle in a VPC is not an optional modifier, but a fundamental and obligatory part of the construction. Consider the VPC *gun down* ('to shoot someone or something so that they fall'), and the raw verb *gun* ('to rev up an engine'); clearly the particle is playing much more than a modifying role.

A better approach would be to make the particle a required part of the construction by building it directly into the verb's subcategorisation frame. In the preceding example, we could conceive the VPC version of the verb to have the category $((S\backslash NP)/NP)/Particle$. The question is then how the particle should be represented. None of the existing atomic categories (N, NP, S, PP) work well in this situation, and all open the door to CCG transformations that would be undesirable in this context. Consequently, we chose to introduce a new tag, RP.

In the simplest case, we have the joined configuration (shown in Figure 1), which requires only functional application. The joined configuration was chosen as the default due to its overwhelming prevalence.

The rarer split case (shown in Figure 2) is slightly more complicated. We use a combination of type-raising and backward-crossed-composition (similar to the Steedman and Baldridge (2006) analysis of heavy noun phrase shift), whilst leaving the verb and particle categories unchanged.

An alternative option for the split case would have been to simply introduce a new category for the verb. However, this approach increases the category ambiguity of the words, and is also opposed to the general design of the formalism, which prefers to handle such surface variation using only combinatory rules.

Finally, we show that our representation can

comfortably accommodate a coordination construction where two verbs share a particle. This relatively rare particle sharing phenomenon occurs only once in the Penn Treebank, and is dealt with in our representation using the simplified coordination combinator, as shown in Figure 3.

One problem with the representation is its tendency to over-generate. English grammar requires that VPCs with a pronominal object be in the split configuration (*she took it away* but not **she took away it*), however this restriction is not observed in our representation, thus allowing invalid sentences. The same applies for manner adverbs occurring between the verb and the particle; English grammar disallows constructs like **they tracked quickly down the thief*, however these are accepted in our proposed representation.

4 Modifying the Corpus

The next stage in our process was modifying CCGbank to accommodate the changes. This involved changing both the syntactic derivations and the word-word dependencies in the predicate-argument structure. The details of the structure of CCGbank can be found in Hockenmaier and Steedman (2005).

To simplify the manipulation of the CCG structures, we first read them into Python as tree-structures, and then wrote these to an external database[1]. This gave us quite a lot of flexibility in querying, retrieving and modifying the structures.

To locate VPCs within the corpus, we relied on a combination of PropBank's (Kingsbury and Palmer, 2002) argument structure labeling and the tags in the Penn Treebank. PropBank provides a listing of every verb (relation) in the corpus, along with its arguments. The word positions for each relation and its arguments are also given, making multiword relations (such as VPCs) readily identifiable. Whenever a multiword relation was found that also contained an RP tag in the Penn Treebank (RP being the Penn Treebank's tag for particles), we took that set of words as being a VPC. This approach errs on the side of caution — there are some valid VPCs in the Penn Treebank that do not have the particle tagged as RP.

A quick survey of the discovered VPCs revealed some interesting features. In total there were 2,578 VPCs. Grouping them based on whether or not the verb and particle share the same parent node in the CCGbank derivation (which loosely equates to the joined-split distinction) yields the results in Table 2. Such a decisive split indicates that there is a definite bias towards the joined configuration, which has the advantage of simplifying the common joined case, but making the split cases even more difficult to identify.

	Same Parent	Different Parents
Count	2425	153
%	94.1%	5.9%

Table 2: Verb and Particle parents in CCGbank

Count	Category
1339	(S\NP)/NP
647	S\NP
302	(S\NP)/PP
96	((S\NP)/PP)/NP
89	(S\NP)/(S\NP)
69	(S\NP)/S
15	((S\NP)/(S\NP))/NP
4	((S\NP)/NP)/PP
3	((S\NP)/PP)/PP
3	N
2	(S\S)\NP

Table 3: Summary of the Verb Categories.

Count	Category
2541	(S\NP)\(S\NP)
10	PP/PP
8	PP/NP
5	(S\NP)/PP
3	S\S
3	N\N
2	((S\NP)\(S\NP))/PP
2	S\NP

Table 4: Summary of the Particle Categories.

Tables 3 and 4 summarise the original CCG categories assigned to the verbs and particles in each VPC that occurred more than once. There is a lot of variation in the tail of each distribution as well as several erroneous categories, although both groups have one category that clearly dominates the rest. The verbs are dominated by the transitive and intransitive categories and the particles are almost exclusively tagged as adverbial modifiers.

For each of the main categories assigned, we hand-crafted a transformation rule to convert instances of that category to our CCG representa-

[1] Acknowledgements to Tim Dawborn for his preparatory work on this system.

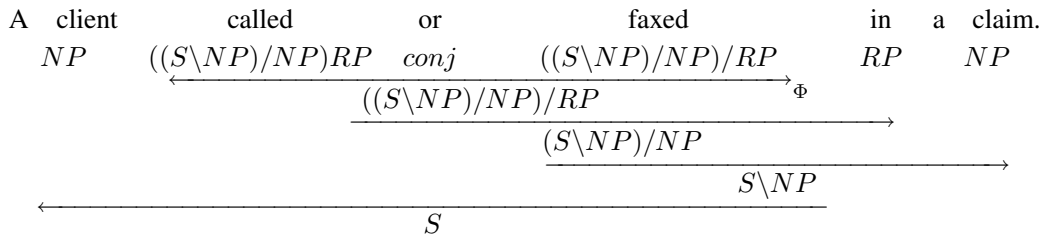

Figure 3: Using the coordination combinator to handle shared particles

Model	LP	LR	LF	LF (POS)	SENT ACC	UP	UR	UF	CAT ACC	COV
C&C	88.06	86.43	87.24	85.25	35.67	93.88	92.13	93.00	94.16	99.06
C&C + VPC	87.90	86.34	87.11	85.11	35.73	93.80	92.13	92.96	94.06	99.06

Table 1: Comparison of results before (top line) and after (bottom line) using the modified VPC corpus.

tion. Instances of VPC nominalisation and categories that occurred with low frequently were left untouched (about 25 instances in total).

5 Results

After modifying the Clark and Curran parser to include support for the new categories that were produced by the conversion process, we retrained the parser on the modified corpus, and then retested it using the same procedure described in Clark and Curran (2007). Our results are shown in Table 1, along with those obtained by Clark and Curran on the unmodified corpus using their hybrid model.

The LP, LR, and LF columns give the labelled precision, recall and F-score respectively for labelled CCG dependencies. We can see that there was a very slight decrease in performance, however considering that the task has been made more difficult by the addition of categories and the resulting parse is structurally and semantically more sound, this is a very small penalty. The statistics for the unlabelled dependencies (UP, UR and UF) show a similar trend. Additionally, as 5.09% of the sentences in the corpus contained VPCs (using our method of detection), we could assume that consistent misclassification would have led to a much larger performance hit.

Table 1 also shows the labelled F-score on automatically assigned POS tags, which also has a similar small performance drop. This is surprising because we expected the preposition/particle distinction to be more challenging for the POS tagger, and that these errors would flow onto the parser.

Table 5 shows the performance of the verb-particle dependencies themselves. There are 97 VPCs in Section 00, and the parser successfully retrieves the vast majority of them, even with automatically assigned POS tags. However, it is far worse at correctly determining the full subcategorisation frame for the verbs, with only 67% of verb categories (65 of 97) being completely correct with gold POS tags.

Type	Frequency
in Gold Standard	97
found by parser (gold POS)	96
found by parser (auto POS)	91
given correct category (gold POS)	65
given correct category (auto POS)	56

Table 5: VPCs in CCGbank Section 00

6 Conclusion

By employing both PropBank and the Penn Treebank, we have been able to produce a modified version of the CCGbank corpus that contains a more syntactically and semantically sound annotation of VPCs. Training the Clark and Curran CCG parser on the new corpus produced equivalent empirical results to the original parser, despite the additional complexity of the augmented corpus. Our initial results demonstrate that VPCs can be parsed efficiently and in a linguistically sophisticated manner using CCG.

7 Acknowledgements

This work was partially supported by the Capital Markets Cooperative Research Centre Limited. We would also like to thank the anonymous reviewers for their helpful comments.

References

Timothy Baldwin and Aline Villavicencio. Extracting the unextractable: a case study on verb-particles. In *Proceedings of the 2002 Conference on Natural Language Learning*, pages 1–7, Taipei, Taiwan, August 2002.

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

Julia Hockenmaier. Data and models for statistical parsing with Combinatory Categorial Grammar. *School of Informatics, Edinburgh, University of Edinburgh*, 280, 2003.

Julia Hockenmaier and Mark Steedman. CCGbank: Users' manual. *Technical Reports (CIS)*, page 52, 2005.

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 and James R Curran. Improving the complement/adjunct distinction in CCGBank. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics (PACLING-07)*, pages 210–217, 2007.

Paul Kingsbury and Martha Palmer. From treebank to propbank. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, pages 1989–1993, 2002.

Paul Kingsbury and Martha Palmer. Propbank: the next level of treebank. In *Proceedings of Treebanks and lexical Theories*, 2003.

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

Mitchell P Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. The Penn Treebank: annotating predicate argument structure. In *HLT '94: Proceedings of the workshop on Human Language Technology*, pages 114–119, Morristown, NJ, USA, 1994. Association for Computational Linguistics. ISBN 1-55860-357-3.

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

Mark Steedman and Jason Baldridge. Combinatory Categorial Grammar. *Encyclopedia of Language and Linguistics*, 2:610–622, 2006.

Aline Villavicencio. Verb-particle constructions and lexical resources. In *Proceedings of the Meeting of the Asociation for Computational Linguistics: 2003 workshop on Multiword expressions*, pages 57–64, Sapporo, Japan, July 2003.