# Unbounded Dependency Recovery for Parser Evaluation

**Laura Rimell** and **Stephen Clark**
University of Cambridge
Computer Laboratory
laura.rimell@cl.cam.ac.uk
stephen.clark@cl.cam.ac.uk

**Mark Steedman**
University of Edinburgh
School of Informatics
steedman@inf.ed.ac.uk

## Abstract

This paper introduces a new parser evaluation corpus containing around 700 sentences annotated with unbounded dependencies, from seven different grammatical constructions. We run a series of off-the-shelf parsers on the corpus to evaluate how well state-of-the-art parsing technology is able to recover such dependencies. The overall results range from 25% accuracy to 59%. These low scores call into question the validity of using Parseval scores as a general measure of parsing capability. We discuss the importance of parsers being able to recover unbounded dependencies, given their relatively low frequency in corpora. We also analyse the various errors made on these constructions by one of the more successful parsers.

## 1 Introduction

Statistical parsers are now obtaining Parseval scores of over 90% on the WSJ section of the Penn Treebank (Bod, 2003; Petrov and Klein, 2007; Huang, 2008; Carreras et al., 2008). McClosky et al. (2006) report an F-score of 92.1% using self-training applied to the reranker of Charniak and Johnson (2005). Such scores, in isolation, may suggest that statistical parsing is close to becoming a solved problem, and that further incremental improvements will lead to parsers becoming as accurate as POS taggers.

A single score in isolation can be misleading, however, for a number of reasons. First, the single score is an aggregate over a highly skewed distribution of all constituent types; evaluations which look at individual constituent or dependency types show that the accuracies on some, semantically important, constructions, such as coordination and PP-attachment, are much lower (Collins, 1999). Second, it is well known that the accuracy of parsers trained on the Penn Treebank degrades when they are applied to different genres and domains (Gildea, 2001). Finally, some researchers have argued that the Parseval metrics (Black et al., 1991) are too forgiving with respect to certain errors and that an evaluation based on syntactic dependencies, for which scores are typically lower, is a better test of parser performance (Lin, 1995; Carroll et al., 1998).

In this paper we focus on the first issue, that the performance of parsers on some constructions is much lower than the overall score. The constructions that we focus on are various unbounded dependency constructions. These are interesting for parser evaluation for the following reasons: one, they provide a strong test of the parser's knowledge of the grammar of the language, since many instances of unbounded dependencies are difficult to recover using shallow techniques in which the grammar is only superficially represented; and two, recovering these dependencies is necessary to completely represent the underlying predicate-argument structure of a sentence, useful for applications such as Question Answering and Information Extraction.

To give an example of the sorts of constructions we are considering, and the (in)ability of parsers to recover the corresponding unbounded dependencies, none of the parsers that we have tested were able to recover the dependencies shown in bold from the following sentences:

*We have also developed techniques for recognizing and locating underground nuclear tests through the* **waves** *in the ground which they* **generate**.

*By Monday , they hope to have a sheaf of* **documents** *both sides can* **trust**.

*By means of charts showing wave-travel times and depths in the ocean at various locations , it is possible to estimate the* **rate** *of approach and probable* **time** *of arrival at Hawaii* **of** *a tsunami getting under way at any spot in the Pacific .*

The contributions of this paper are as follows. First, we present the first set of results for the recovery of a variety of unbounded dependencies, for a range of existing parsers. Second, we describe the creation of a publicly available unbounded dependency test suite, and give statistics summarising properties of these dependencies in naturally occurring text. Third, we demonstrate that performing the evaluation is surprisingly difficult, because of different conventions across the parsers as to how the underlying grammar is represented. Fourth, we show that current parsing technology is very poor at representing some important elements of the argument structure of sentences, and argue for a more focused construction-based parser evaluation as a complement to existing grammatical relation-based evaluations. We also perform an error-analysis for one of the more successful parsers.

There has been some prior work on evaluating parsers on long-range dependencies, but no work we are aware of that has the scope and focus of this paper. Clark et al. (2004) evaluated a CCG parser on a small corpus of object extraction cases. Johnson (2002) began the body of work on inserting traces into the output of Penn Treebank (PTB) parsers, followed by Levy and Manning (2004), among others. This PTB work focused heavily on the representation in the Treebank, evaluating against patterns in the trace annotation. In this paper we have tried to be more "formalism-independent" and construction focused.

## 2 Unbounded Dependency Corpus

### 2.1 The constructions

An unbounded dependency construction contains a word or phrase which appears to have been moved, while being interpreted in the position of the resulting "gap". An unlimited number of clause boundaries may intervene between the moved element and the gap (hence "unbounded").

The seven constructions in our corpus were chosen for being relatively frequent in text, compared to other unbounded dependency types, and relatively easy to identify. An example of each construction, along with its associated dependencies, is shown in Table 1. Here we give a brief description of each construction.

**Object extraction from a relative clause** is characterised by a relative pronoun (a *wh*-word or *that*) introducing a clause from which an argument in object position has apparently been extracted: *the paper which I wrote*. Our corpus includes cases where the extracted word is (semantically) the object of a preposition in the verb phrase: *the agency that I applied to*.

**Object extraction from a reduced relative clause** is essentially the same, except that there is no overt relative pronoun: *the paper I wrote; the agency I applied to*. We did not include participial reduced relatives such as *the paper written by the professor*.

**Subject extraction from a relative clause** is characterised by the apparent extraction of an argument from subject position: *the instrument that measures depth*. A relative pronoun is obligatory in this construction. Our corpus includes passive subjects: *the instrument which was used by the professor*.

**Free relatives** contain relative pronouns without antecedents: *I heard what she said*, where *what* does not refer to any other noun in the sentence. Free relatives can usually be paraphrased by noun phrases such as *the thing she said* (a standard diagnostic for distinguishing them from embedded interrogatives like *I wonder what she said*). The majority of sentences in our corpus are object free relatives, but we also included some adverbial free relatives: *She told us how to do it*.

**Object *wh*-questions** are questions in which the *wh*-word is the semantic object of the verb: *What did you eat?*. Objects of prepositions are included: *What city does she live in?*. Also included are a few cases where the *wh*-word is arguably adverbial, but is selected for by the verb: *Where is the park located?*.

**Right node raising (RNR)** is characterised by coordinated phrases from which a shared element apparently moves to the right: *Mary saw and Susan bought the book*. This construction is unique within our corpus in that the "raised" element can have a wide variety of grammatical functions. Examples include: noun phrase object of verb, noun phrase object of preposition (*material about or messages from the communicator*), a combination of the two (*applied for and won approval*), prepositional phrase modifier (*president and chief executive of the company*), infinitival modifier (*the will and the capacity to prevent the event*), and modified noun (*a good or a bad decision*).

**Subject extraction from an embedded clause** is characterised by a semantic subject which is ap-

**Object extraction from a relative clause**

Each must match Wisman's "pie" with the **fragment** that he **carries** with him.

```
dobj(carries, fragment)
```

**Object extraction from a reduced relative clause**

Put another way, the decline in the yield suggests stocks have gotten pretty rich in price relative to the **dividends** they **pay**, some market analysts say.

```
dobj(pay, dividends)
```

**Subject extraction from a relative clause**

It consists of a **series** of pipes and a pressure-measuring **chamber** which **record** the rise and fall of the water surface.

```
nsubj(record, series)
nsubj(record, chamber)
```

**Free relative**

He tried to ignore **what** his own common sense **told** him, but it wasn't possible; her motives were too blatant.

```
dobj(told, what)
```

**Object *wh*-question**

What **city** does the Tour de France end **in**?

```
pobj(in, city)
```

**Right node raising**

For the third year in a row, consumers voted Bill Cosby **first** and James Garner **second in** persuasiveness as spokesmen in TV commercials, according to Video Storyboard Tests, New York.

```
prep(first, in)
prep(second, in)
```

**Subject extraction from an embedded clause**

In assigning to God the **responsibility** which he learned could not **rest** with his doctors, Eisenhower gave evidence of that weakening of the moral intuition which was to characterize his administration in the years to follow.

```
nsubj(rest, responsibility)
```

Table 1: Examples of the seven constructions in the unbounded dependency corpus.

parently extracted across two clause boundaries, as shown in the following bracketing (where $*$ marks the origin of the extracted element): *the responsibility which [the government said [$*$ lay with the voters]].* Our corpus includes sentences where the embedded clause is a so-called small clause, i.e. one with a null copula verb: *the plan that she considered foolish*, where *plan* is the semantic subject of *foolish.*

## 2.2 The data

The corpus consists of approximately 100 sentences for each of the seven constructions; 80 of these were reserved for each construction for testing, giving a test set of 560 sentences in total, and the remainder were used for initial experimentation (for example to ensure that default settings for the various parsers were appropriate for this data). We did not annotate the full sentences, since we are only interested in the unbounded dependencies and full annotation of such a corpus would be extremely time-consuming.

With the exception of the question construction, all sentences were taken from the PTB, with roughly half from the WSJ sections (excluding 2-21 which provided the training data for many of the parsers in our set) and half from Brown (roughly balanced across the different sections). The questions were taken from the question data in Rimell and Clark (2008), which was obtained from various years of the TREC QA track. We chose to use the PTB as the main source because the use of traces in the PTB annotation provides a starting point for the identification of unbounded dependencies.

Sentences were selected for the corpus by a combination of automatic and manual processes. A regular expression applied to PTB trees, searching for appropriate traces for a particular construction, was first used to extract a set of candidate sentences. All candidates were manually reviewed and, if selected, annotated with one or more grammatical relations representing the relevant unbounded dependencies in the sentence. Some of the annotation in the treebank makes identification of some constructions straightforward; for example right node raising is explicitly represented as RNR. Indeed it may have been possible to fully automate this process with use of the `tgrep` search tool. However, in order to obtain reliable statistics regarding frequency of occurrence, and to ensure a high-quality resource, we used fairly broad regular expressions to identify the original set followed by manual review.

We chose to represent the dependencies as grammatical relations (GRs) since this format seemed best suited to represent the kind of semantic relationship we are interested in. GRs are head-based dependencies that have been suggested as a more appropriate representation for general parser evaluation than phrase-structure trees (Carroll et al., 1998). Table 1 gives examples of how GRs are used to represent the relevant dependencies. The particular GR scheme we used was based on the Stanford scheme (de Marneffe et al., 2006); however, the specific GR scheme is not too crucial since the whole sentence is not being represented in the corpus, only the unbounded dependencies.

## 3 Experiments

The five parsers described in Section 3.2 were used to parse the test sentences in the corpus, and the percentage of dependencies in the test set recovered by each parser for each construction was calculated. The details of how the parsers were run and how the parser output was matched against the gold standard are given in Section 3.3. This is essentially a recall evaluation, and so is open to abuse; for example, a program which returns all the possible word pairs in a sentence, together with all possible labels, would score 100%. However, this is easily guarded against: we simply assume that each parser is being run in a "standard" mode, and that each parser has already been evaluated on a full corpus of GRs in order to measure precision and recall across all dependency types. (Calculating precision for the unbounded dependency evaluation would be difficult since that would require us to know how many *incorrect* unbounded dependencies were returned by each parser.)

| Construction | WSJ | Brown | Overall |
|---|---|---|---|
| Obj rel clause | 2.3 | 1.1 | 1.4 |
| Obj reduced rel | 2.7 | 2.8 | 2.8 |
| Sbj rel clause | 10.1 | 5.7 | 7.4 |
| Free rel | 2.6 | 0.9 | 1.3 |
| RNR | 2.2 | 0.9 | 1.2 |
| Sbj embedded | 2.0 | 0.3 | 0.4 |

Table 2: Frequency of constructions in the PTB (percentage of sentences).

### 3.1 Statistics relating to the constructions

Table 2 shows the percentage of sentences in the PTB, from those sections that were examined, which contain an example of each type of unbounded dependency. Perhaps not surprisingly, root subject extractions from relative clauses are by far the most common, with the remaining constructions occurring in roughly between 1 and 2% of sentences. Note that, although examples of each individual construction are relatively rare, the combined total is over 10% (assuming that each construction occurs independently). Section 6 contains a discussion regarding the frequency of occurrence of these events and the consequences of this for parser performance.

Table 3 shows the average and maximum distance between head and dependent for each construction, as measured by the difference between word indices. This is a fairly crude measure of distance but gives some indication of how "long-range" the dependencies are for each construction. The cases of object extraction from a relative clause and subject extraction from an embedded clause provide the longest dependencies, on average. The following sentence gives an example of how far apart the head and dependent can be in a

| Construction | Avg Dist | Max Dist |
|---|---|---|
| Obj rel clause | 6.8 | 21 |
| Obj reduced rel | 3.4 | 8 |
| Sbj rel clause | 4.4 | 18 |
| Free rel | 3.4 | 16 |
| Obj wh-question | 4.8 | 9 |
| RNR | 4.8 | 23 |
| Sbj embedded | 7.0 | 21 |

Table 3: Distance between head and dependent.

subject embedded construction:

*the same* **stump** *which had impaled the car of many a guest in the past thirty years and which he refused to have* **removed**.

## 3.2 The parsers

The parsers that we chose to evaluate are the C&C CCG parser (Clark and Curran, 2007), the Enju HPSG parser (Miyao and Tsujii, 2005), the RASP parser (Briscoe et al., 2006), the Stanford parser (Klein and Manning, 2003), and the DCU post-processor of PTB parsers (Cahill et al., 2004), based on LFG and applied to the output of the Charniak and Johnson reranking parser. Of course we were unable to evaluate every publicly available parser, but we believe these are representative of current wide-coverage robust parsing technology.[1]

The C&C parser is based on CCGbank (Hockenmaier and Steedman, 2007), a CCG version of the Penn Treebank. It is ideally suited for this evaluation because CCG was designed to capture the unbounded dependencies being considered. The Enju parser was designed with a similar motivation to C&C, and is also based on an automatically extracted grammar derived from the PTB, but the grammar formalism is HPSG rather than CCG. Both parsers produce head-word dependencies reflecting the underlying predicate-argument structure of a sentence, and so in theory should be straightforward to evaluate.

The RASP parser is based on a manually constructed POS tag-sequence grammar, with a statistical parse selection component and a robust partial-parsing technique which allows it to return output for sentences which do not obtain a full spanning analysis according to the grammar. RASP has not been designed to capture many of the dependencies in our corpus; for example, the tag-sequence grammar has no explicit representation of verb subcategorisation, and so may not know that there is a missing object in the case of extraction from a relative clause (though it does recover some of these dependencies). However, RASP is a popular parser used in a number of applications, and it returns dependencies in a suitable format for evaluation, and so we considered it to be an appropriate and useful member of our parser set.

The Stanford parser is representative of a large number of PTB parsers, exemplified by Collins (1997) and Charniak (2000). The Parseval scores reported for the Stanford parser are not the highest in the literature, but are competitive enough for our purposes. The advantage of the Stanford parser is that it returns dependencies in a suitable format for our evaluation. The dependencies are obtained by a set of manually defined rules operating over the phrase-structure trees returned by the parser (de Marneffe et al., 2006). Like RASP, the Stanford parser has not been designed to capture unbounded dependencies; in particular it does not make use of any of the trace information in the PTB. However, we wanted to include a "standard" PTB parser in our set to see which of the unbounded dependency constructions it is able to deal with.

Finally, there is a body of work on inserting trace information into the output of PTB parsers (Johnson, 2002; Levy and Manning, 2004), which is the annotation used in the PTB for representing unbounded dependencies. The work which deals with the PTB representation directly, such as Johnson (2002), is difficult for us to evaluate because it does not produce explicit dependencies. However, the DCU post-processor is ideal because it does produce dependencies in a GR format. It has also obtained competitive scores on general GR evaluation corpora (Cahill et al., 2004).

## 3.3 Parser evaluation

The parsers were run essentially out-of-the-box when parsing the test sentences. The one exception was C&C, which required some minor adjusting of parameters, as described in the parser documentation, to obtain close to full coverage on the data. In addition, the C&C parser comes with a

[1]One obvious omission is any form of dependency parser (McDonald et al., 2005; Nivre and Scholz, 2004). However, the dependencies returned by these parsers are local, and it would be non-trivial to infer from a series of links whether a long-range dependency had been correctly represented. Also, dependency parsers are not significantly better at recovering head-based dependencies than constituent parsers based on the PTB (McDonald et al., 2005).

|  | Obj RC | Obj Red | Sbj RC | Free | Obj Q | RNR | Sbj Embed | Total |
|---|---|---|---|---|---|---|---|---|
| C&C | **59.3** | 62.6 | 80.0 | 72.6 | (**81.2**) 27.5 | **49.4** | 22.4 | (**59.7**) 53.6 |
| Enju | 47.3 | **65.9** | **82.1** | **76.2** | 32.5 | 47.1 | **32.9** | **54.4** |
| DCU | 23.1 | 41.8 | 56.8 | 46.4 | 27.5 | 40.8 | 5.9 | 35.7 |
| Rasp | 16.5 | 1.1 | 53.7 | 17.9 | 27.5 | 34.5 | 15.3 | 25.3 |
| Stanford | 22.0 | 1.1 | 74.7 | 64.3 | **41.2** | 45.4 | 10.6 | 38.1 |

Table 4: Parser accuracy on the unbounded dependency corpus; the highest score for each construction is in bold; the figures in brackets for C&C derive from the use of a separate question model.

specially designed question model, and so we applied both this and the standard model to the object *wh*-question cases.

The parser output was evaluated against each dependency in the corpus. Due to the various GR schemes used by the parsers, an exact match on the dependency label could not always be expected. We considered a correctly recovered dependency to be one where the gold-standard head and dependent were correctly identified, and the label was an "acceptable match" to the gold-standard label. To be an acceptable match, the label had to indicate the grammatical function of the extracted element at least to the level of distinguishing active subjects, passive subjects, objects, and adjuncts. For example, we allowed an `obj` (object) relation as a close enough match for `dobj` (direct object) in the corpus, even though `obj` does not distinguish different kinds of objects, but we did not allow generic "relative pronoun" relations that are underspecified for the grammatical role of the extracted element.

The differences in GR schemes were such that we ended up performing a time-consuming largely manual evaluation. We list here some of the key differences that made the evaluation difficult.

In some cases, the parser's set of labels was less fine-grained than the gold standard. For example, RASP represents the direct objects of both verbs and prepositions as `dobj` (direct object), whereas the gold-standard uses `pobj` for the preposition case. We counted the RASP output as correctly matching the gold standard.

In other cases, the label on the dependency containing the gold-standard head and dependent was too underspecified to be acceptable by itself. For example, where the gold-standard relation was `dobj(placed,buckets)`, DCU produced `relmod(buckets,placed)` with a generic "relative modifier" label. However, the correct label could be recovered from elsewhere in the parser output, specifically a combination of `relpro(buckets,which)` and `obj(placed,which)`. In this case we counted the DCU output as correctly matching the gold standard.

In some constructions the Stanford scheme, upon which the gold-standard was based, makes different choices about heads than other schemes. For example, in the the phrase *Honolulu, which is the center of the warning system*, the corpus contains a subject dependency with *center* as the head: `nsubj(center,Honolulu)`. Other schemes, however, treat the auxiliary verb *is* as the head of the dependency, rather than the predicate nominal *center*. As long as the difference in head selection was due solely to the idiosyncracies of the GR schemes involved, we counted the relation as correct.

Finally, the different GR schemes treat coordination differently. In the corpus, coordinated elements are always represented with two dependencies. Thus the phrase *they may half* **see** *and half* **imagine** *the old* **splendor** has two gold-standard dependencies: `dobj(see,splendor)` and `dobj(imagine,splendor)`. If a parser produced only the former dependency, but appeared to have the coordination correct, then we awarded two marks, even though the second dependency was not explicitly represented.

## 4 Results

Accuracies for the various parsers are shown in Table 4, with the highest score for each construction in bold. Enju and C&C are the top performers, operating at roughly the same level of accuracy across most of the constructions. Use of the C&C question model made a huge difference for the *wh*-object construction (81.2% vs. 27.5%), showing that adaptation techniques specific to a particular construction can be successful (Rimell and Clark, 2008).

In order to learn more from these results, in Section 5 we analyse the various errors made by the C&C parser on each construction. The conclusions that we arrive at for the C&C parser we would also expect to apply to Enju, on the whole, since the design of the two parsers is so similar. In fact, some of the recommendations for improvement on this corpus, such as the need for a better parsing model to make better attachment decisions, are parser independent.

The poor performance of RASP on this corpus is clearly related to a lack of subcategorisation information, since this is crucial for recovering extracted arguments. For Stanford, incorporating the trace information from the PTB into the statistical model in some way is likely to help. The C&C and Enju parsers do this through their respective grammar formalisms. Our informal impression of the DCU post-processor is that it has much of the machinery available to recover the dependencies that the Enju and C&C parsers do, but for some reason which is unclear to us it performs much worse.

## 5 Analysis of the C&C Parser

We categorised the errors made by the C&C parser on the development data for each construction. We chose the C&C parser for the analysis because it was one of the top performers and we have more knowledge of its workings than those of Enju.

The C&C parser first uses a supertagger to assign a small number of CCG lexical categories (essentially subcategorisation frames) to each word in the sentence. These categories are then combined using a set of combinatory rules to build a CCG derivation. The parser uses a log-linear probability model to select the highest-scoring derivation (Clark and Curran, 2007). In general, errors in dependency recovery may occur if the correct lexical category is not assigned by the supertagger for one or more of the words in a sentence, or if an incorrect derivation is chosen by the parsing model.

For unbounded dependency recovery, one source of errors (labeled **type 1** in Table 5) is the wrong lexical category being assigned to the word (normally a verb or preposition) governing the extraction site. In *these testaments that I would submit here*, if *submit* is assigned a category for an intransitive rather than transitive verb, the verb-object relation will not be recovered.

| | 1a | 1b | 1c | 1d | 2 | 3 | Errs | Tot |
|---|---|---|---|---|---|---|---|---|
| ObjRC | | | 6 | | 5 | 2 | 13 | 20 |
| ObjRed | 2 | | 1 | 1 | 1 | 3 | 8 | 23 |
| SbjRC | | | | | 8 | 1 | 9 | 43 |
| Free | 1 | | | | | 1 | 2 | 22 |
| ObjQ | | | 2 | 2 | | | 4 | 25 |
| RNR | | | 2 | 1 | 7 | 3 | 13 | 28 |
| SbjEmb | 3 | 2 | 1 | | | 4 | 10 | 13 |
| Subtotal | 6 | 2 | 12 | 4 | | | | |
| Total | 24 | | | | 21 | 14 | 59 | 174 |

Table 5: Error analysis for C&C. *Errs* is the total number of errors for a construction, *Tot* is the number of dependencies of that type in the development data.

There are a number of reasons why the wrong category may be assigned. First, the lexicon may not contain enough information about possible categories for the word (**1a**), or the necessary category may not exist in the parser's grammar at all (**1b**). Even if the grammar contains the correct category and the lexicon makes it available, the parsing model may not choose it (**1c**). Finally, a POS-tagging error on the word may mislead the parser into assigning the wrong category (**1d**).[2]

A second source of errors (**type 2**) is attachment decisions that the parser makes independently of the unbounded dependency. In *Morgan . . . carried in several buckets of water from the spring which he poured into the copper boiler*, the parser assigns the correct categories for the relative pronoun and verb, but chooses *spring* rather than *buckets* as the head of the relativized NP (i.e. the object of *pour*). Most attachment errors involve prepositional phrases (PPs) and coordination, which have long been known to be areas where parsers need improvement.

Finally, errors in unbounded dependency recovery may be due to complex errors in the surrounding parse context (**type 3**). We will not comment more on these cases since they do not tell us much about unbounded dependencies in particular.

Table 5 shows the distribution of error types across constructions for the C&C parser. Subject relative clauses, for example, did not have any errors of type 1, because a verb with an extracted

[2]We considered an error to be type 1 only when the category error occurred on the word governing the extraction site, except in the subject embedded sentences, where we also included the embedding verb, since the category of this verb is key to dependency recovery.

subject does not require a special lexical category. Most of the errors here are of type 2. For example, in *a series of pipes and a pressure-measuring chamber which record the rise and fall of the water surface*, the parser attaches the relative clause to *chamber* but not to *series*.

Subject embedded sentences show a different pattern. Many of the errors can be attributed to problems with the lexicon and grammar (1a and 1b). For example, in *shadows that they imagined were Apaches*, the word *imagined* never appears in the training data with the correct category, and so the required entry is missing from the lexicon.

Object extraction from a relative clause had a higher number of errors involving the parsing model (1c). In *the first carefree, dreamless sleep that she had known*, the transitive category is available for *known*, but not selected by the model.

The majority of the errors made by the parser are due to insufficient grammar coverage or weakness in the parsing model due to sparsity of head dependency data, the same fundamental problems that have dogged automatic parsing since its inception. Hence one view of statistical parsing is that it has allowed us to solve the easy problems, but we are still no closer to a general solution for the recovery of the "difficult" dependencies. One possibility is to create more training data targeting these constructions – effectively "active learning by construction" – in the way that Rimell and Clark (2008) were able to build a question parser. We leave this idea for future work.

## 6 Discussion

Unbounded dependencies are rare events, out in the Zipfian "long tail". They will always constitute a fraction of a percent of the overall total of head-dependencies in any corpus, a proportion too small to make a significant impact on global measures of parser accuracy, when expressive parsers are compared to those that merely approximate human grammar using finite-state or context-free covers. This will remain the case even when such measures are based on dependencies, rather than on parse trees.

Nevertheless, unbounded dependencies remain highly significant in a much more important sense. They support the constructions that are central to those applications of parsing technology for which precision is as important as recall, such as open-domain question-answering. As low-power approximate parsing methods improve (as they must if they are ever to be usable at all for such tasks), we predict that the impact of the constructions we examine here will become evident. No matter how infrequent object questions like "What do frogs eat?" are, if they are answered as if they were subject questions ("Herons"), users will rightly reject any excuse in terms of the overall statistical distribution of related bags of words.

Whether such improvements in parsers come from the availability of more human-labeled data, or from a breakthrough in unsupervised machine learning, we predict an imminent "Uncanny Valley" in parsing applications, due to the inability of parsers to recover certain semantically important dependencies, of the kind familiar from humanoid robotics and photorealistic animation. In such applications, the closer the superficial resemblance to human behavior gets, the more disturbing subtle departures become, and the more they induce mistrust and revulsion in the user.

## 7 Conclusion

In this paper we have demonstrated that current parsing technology is poor at recovering some of the unbounded dependencies which are crucial for fully representing the underlying predicate-argument structure of a sentence. We have also argued that correct recovery of such dependencies will become more important as parsing technology improves, despite the relatively low frequency of occurrence of the corresponding grammatical constructions. We also see this more focused parser evaluation methodology — in this case construction-focused — as a way of improving parsing technology, as an alternative to the exclusive focus on incremental improvements in overall accuracy measures such as Parseval.

## Acknowledgments

Laura Rimell and Stephen Clark were supported by EPSRC grant EP/E035698/1. Mark Steedman was supported by EU IST Cognitive Systems grant IP FP6-2004-IST-4-27657 (PACO-PLUS). We would like to thank Aoife Cahill for producing the DCU data.

## References

E. Black, S. Abney, D. Flickenger, C. Gdaniec, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski. 1991. A procedure for quantitatively comparing the syntactic coverage of English grammars. In *HLT '91: Proceedings of the Workshop on Speech and Natural Language*, pages 306–311.

Rens Bod. 2003. An efficient implementation of a new DOP model. In *Proceedings of the 10th Meeting of the EACL*, pages 19–26, Budapest, Hungary.

Ted Briscoe, John Carroll, and Rebecca Watson. 2006. The second release of the RASP system. In *Proceedings of the Interactive Demo Session of COLING/ACL-06*, Sydney, Australia.

A. Cahill, M. Burke, R. O'Donovan, J. van Genabith, and A. Way. 2004. Long-distance dependency resolution in automatically acquired wide-coverage PCFG-based LFG approximations. In *Proceedings of the 42nd Meeting of the ACL*, pages 320–327, Barcelona, Spain.

Xavier Carreras, Michael Collins, and Terry Koo. 2008. Dynamic programming and the perceptron for efficient, feature-rich parsing. In *Proceedings of the Twelfth Conference on Natural Language Learning (CoNLL-08)*, pages 9–16, Manchester, UK.

John Carroll, Ted Briscoe, and Antonio Sanfilippo. 1998. Parser evaluation: a survey and a new proposal. In *Proceedings of the 1st LREC Conference*, pages 447–454, Granada, Spain.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Meeting of the ACL*, pages 173–180, Michigan, Ann Arbor.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st Meeting of the NAACL*, pages 132–139, Seattle, WA.

Stephen Clark and James R. Curran. 2007. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4):493–552.

Stephen Clark, Mark Steedman, and James R. Curran. 2004. Object-extraction and question-parsing using CCG. In *Proceedings of the EMNLP Conference*, pages 111–118, Barcelona, Spain.

Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Meeting of the ACL*, pages 16–23, Madrid, Spain.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th LREC Conference*, Genoa, Italy.

Daniel Gildea. 2001. Corpus variation and parser performance. In *Proceedings of the 2001 EMNLP Conference*, Pittsburgh, PA.

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

Liang Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *Proceedings of the 46th Meeting of the ACL*, pages 586–594, Columbus, Ohio.

Mark Johnson. 2002. A simple pattern-matching algorithm for recovering empty nodes and their antecedents. In *Proceedings of the 40th Meeting of the ACL*, pages 136–143, Philadelphia, PA.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of Association for Computational Linguistics*, pages 423–430, Sapporo, Japan.

Roger Levy and Christopher Manning. 2004. Deep dependencies from context-free statistical parsers: correcting the surface dependency approximation. In *Proceedings of the 42nd Meeting of the ACL*, pages 328–335, Barcelona, Spain.

Dekang Lin. 1995. A dependency-based method for evaluating broad-coverage parsers. In *Proceedings of IJCAI-95*, pages 1420–1425, Montreal, Canada.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the North American Chapter of the Association for Computational Linguistics Conference*, pages 152–159, Brooklyn, NY.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Meeting of the ACL*, pages 91–98, Michigan, Ann Arbor.

Yusuke Miyao and Jun'ichi Tsujii. 2005. Probabilistic disambiguation models for wide-coverage HPSG parsing. In *Proceedings of the 43rd Meeting of the ACL*, pages 83–90, Michigan, Ann Arbor.

J. Nivre and M. Scholz. 2004. Deterministic dependency parsing of English text. In *Proceedings of COLING-04*, pages 64–70, Geneva, Switzerland.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of the HLT/NAACL Conference*, Rochester, NY.

Laura Rimell and Stephen Clark. 2008. Adapting a lexicalized-grammar parser to contrasting domains. In *Proceedings of the 2008 EMNLP Conference*, pages 475–484, Honolulu, Hawai'i.