# Verb Phrase Ellipsis detection using Automatically Parsed Text

**Leif Arda Nielsen**
Department of Computer Science
King's College London
nielsen@dcs.kcl.ac.uk

## Abstract

This paper describes a Verb Phrase Ellipsis (VPE) detection system, built for robustness, accuracy and domain independence. The system is corpus-based, and uses a variety of machine learning techniques on free text that has been automatically parsed using two different parsers. Tested on a mixed corpus comprising a range of genres, the system achieves a 72% F1-score. It is designed as the first stage of a complete VPE resolution system that is input free text, detects VPEs, and proceeds to find the antecedents and resolve them.

## 1 Introduction

Ellipsis is a linguistic phenomenon that has received considerable attention, mostly focusing on its interpretation. Most work on ellipsis (Fiengo and May, 1994; Lappin, 1993; Dalrymple et al., 1991; Kehler, 1993; Shieber et al., 1996) is aimed at discerning the procedures and the level of language processing at which ellipsis resolution takes place, or focuses on ambiguous and difficult cases. The detection of elliptical sentences or the identification of the antecedent and elided clauses within them are usually not dealt with, but taken as given. Noisy or missing input, which is unavoidable in NLP applications, is not dealt with, and neither is focusing on specific domains or applications. It therefore becomes clear that a robust, trainable approach is needed.

An example of Verb Phrase Ellipsis (VPE), which is detected by the presence of an auxiliary verb without a verb phrase, is seen in example 1. VPE can also occur with semi-auxiliaries, as in example 2.

(1) John$_3$ {loves his$_3$ wife}$_2$. Bill$_3$ does$_1$ too.

(2) But although he was terse, he didn't {rage at me}$_2$ the way I expected him to$_1$.

Several steps of work need to be done for ellipsis resolution :

1. Detecting ellipsis occurrences. First, elided verbs need to be found.
2. Identifying antecedents. For most cases of ellipsis, copying of the antecedent clause is enough for resolution (Hardt, 1997).
3. Resolving ambiguities. For cases where ambiguity exists, a method for generating the full list of possible solutions, and suggesting the most likely one is needed.

This paper describes the work done on the first stage, the detection of elliptical verbs. First, previous work done on tagged corpora will be summarised. Then, new work on parsed corpora will be presented, showing the gains possible through sentence-level features. Finally, experiments using unannotated data that is parsed using an automatic parser are presented, as our aim is to produce a stand-alone system.

We have chosen to concentrate on VP ellipsis due to the fact that it is far more common than other forms of ellipsis, but pseudo-gapping, an example of which is seen in example 3, has also been included due to the similarity of its resolution to VPE (Lappin, 1996). *Do so/it/that* and *so doing* anaphora are not handled, as their resolution is different from that of VPE (Kehler and Ward, 1999).

(3) John writes plays, and Bill does novels.

## 2 Previous work

Hardt's (1997) algorithm for detecting VPE in the Penn Treebank (see Section 3) achieves recall levels of 53% and precision of 44%, giving an F1[1] of 48%, using a simple search tech-

[1]Precision, recall and F1 are defined as :

$$\textit{Recall} = \frac{No(\text{correct ellipses found})}{No(\text{all ellipses in test})} \quad (1)$$

nique, which relies on the parse annotation having identified empty expressions correctly.

In previous work (Nielsen, 2003a; Nielsen, 2003b) we performed experiments on the British National Corpus using a variety of machine learning techniques. These earlier results are not directly comparable to Hardt's, due to the different corpora used. The expanded set of results are summarised in Table 1, for Transformation Based Learning (TBL) (Brill, 1995), GIS based Maximum Entropy Modelling (GIS-MaxEnt)[2] (Ratnaparkhi, 1998), L-BFGS based Maximum Entropy Modelling (L-BFGS-MaxEnt)[3] (Malouf, 2002), Decision Tree Learning (Quinlan, 1993) and Memory Based Learning (MBL) (Daelemans et al., 2002).

| Algorithm | Recall | Precision | F1 |
|---|---|---|---|
| TBL | 69.63 | 85.14 | 76.61 |
| Decision Tree | 60.93 | 79.39 | 68.94 |
| MBL | 72.58 | 71.50 | 72.04 |
| GIS-MaxEnt | 71.72 | 63.89 | 67.58 |
| L-BFGS-MaxEnt | 71.93 | 80.58 | 76.01 |

Table 1: Comparison of algorithms

For all of these experiments, the training features consisted of lexical forms and Part of Speech (POS) tags of the words in a three word forward/backward window of the auxiliary being tested. This context size was determined empirically to give optimum results, and will be used throughout this paper. L-BFGS-MaxEnt uses Gaussian Prior smoothing optimized for the BNC data, while GIS-MaxEnt has a simple smoothing option available, but this deteriorates results and is not used. Both maximum entropy models were experimented with to determine thresholds for accepting results as VPE; GIS-MaxEnt was set to a 20% confidence threshold and L-BFGS-MaxEnt to 35%. MBL was used with its default settings.

While TBL gave the best results, the software we used (Lager, 1999) ran into memory problems and proved problematic with larger datasets. Decision trees, on the other hand, tend to oversimplify due to the very sparse nature of ellipsis, and produce a single rule that classifies everything as non-VPE. This leaves Maximum Entropy and MBL for further experiments.

## 3 Corpus description

The British National Corpus (BNC) (Leech, 1992) is annotated with POS tags, using the CLAWS-4 tagset. A range of sections of the BNC, containing around 370k words[4] with 645 samples of VPE was used as training data. The separate development data consists of around 74k words[5] with 200 samples of VPE.

The Penn Treebank (Marcus et al., 1994) has more than a hundred phrase labels, and a number of empty categories, but uses a coarser tagset. A mixture of sections from the Wall Street Journal and Brown corpus were used. The training section[6] consists of around 540k words and contains 522 samples of VPE. The development section[7] consists of around 140k words and contains 150 samples of VPE.

## 4 Experiments using the Penn Treebank

To experiment with what gains are possible through the use of more complex data such as parse trees, the Penn Treebank is used for the second round of experiments. The results are presented as new features are added in a cumulative fashion, so each experiment also contains the data contained in those before it; the close to punctuation experiment contains the words and POS tags from the experiment before it, the next experiment contains all of these plus the heuristic baseline and so on.

### Words and POS tags

The Treebank, besides POS tags and category headers associated with the nodes of the parse tree, includes empty category information. For the initial experiments, the empty category information is ignored, and the words and POS tags are extracted from the trees. The results in Table 2 are seen to be considerably poorer than those for BNC, despite the comparable data sizes. This can be accounted for by the coarser tagset employed.

$$Precision = \frac{No(\text{correct ellipses found})}{No(\text{all ellipses found})} \quad (2)$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (3)$$

[2]Downloadable from https://sourceforge.net/projects/maxent/

[3]Downloadable from http://www.nlplab.cn/zhangle/maxent_toolkit.html

[4]Sections CS6, A2U, J25, FU6, H7F, HA3, A19, A0P, G1A, EWC, FNS, C8T

[5]Sections EDJ, FR3

[6]Sections WSJ 00, 01, 03, 04, 15, Brown CF, CG, CL, CM, CN, CP

[7]Sections WSJ 02, 10, Brown CK, CR

| Algorithm | Recall | Precision | F1 |
|---|---|---|---|
| MBL | 50.98 | 61.90 | 55.91 |
| GIS-MaxEnt | 34.64 | 79.10 | 48.18 |
| L-BFGS-MaxEnt | 60.13 | 76.66 | 67.39 |

Table 2: Initial results with the Treebank

### Close to punctuation

A very simple feature, that checks for auxiliaries close to punctuation marks was tested. Table 3 shows the performance of the feature itself, characterised by very low precision, and results obtained by using it. It gives a 3% increase in F1 for GIS-MaxEnt, but a 1.5% decrease for L-BFGS-MaxEnt and 0.5% decrease for MBL.

This brings up the point that the individual success rate of the features will not be in direct correlation with gains in overall results. Their contribution will be high if they have high precision for the cases they are meant to address, and if they produce a different set of results from those already handled well, complementing the existing features. Overlap between features can be useful to have greater confidence when they agree, but low precision in the feature can increase false positives as well, decreasing performance. Also, the small size of the development set can contribute to fluctuations in results.

| Algorithm | Recall | Precision | F1 |
|---|---|---|---|
| close-to-punctuation | 30.06 | 2.31 | 4.30 |
| MBL | 50.32 | 61.60 | 55.39 |
| GIS-MaxEnt | 37.90 | 79.45 | 51.32 |
| L-BFGS-MaxEnt | 57.51 | 76.52 | 65.67 |

Table 3: Effects of using the close-to-punctuation feature

### Heuristic Baseline

A simple heuristic approach was developed to form a baseline using only POS data. The method takes all auxiliaries as possible candidates and then eliminates them using local syntactic information in a very simple way. It searches forwards within a short range of words, and if it encounters any other verbs, adjectives, nouns, prepositions, pronouns or numbers, classifies the auxiliary as not elliptical. It also does a short backwards search for verbs. The forward search looks 7 words ahead and the backwards search 3. Both skip 'asides', which are taken to be snippets between commas without verbs in them, such as : "... papers do, however, show ...". This feature gives a 3.5 - 4.5% improvement (Table 4).

| Algorithm | Recall | Precision | F1 |
|---|---|---|---|
| heuristic | 48.36 | 27.61 | 35.15 |
| MBL | 55.55 | 65.38 | 60.07 |
| GIS-MaxEnt | 43.13 | 78.57 | 55.69 |
| L-BFGS-MaxEnt | 62.09 | 77.86 | 69.09 |

Table 4: Effects of using the heuristic feature

```
(SINV
   (ADVP-PRD-TPC-2   (RB so) )
   (VP  (VBZ is)
       (ADVP-PRD (-NONE- *T*-2)  ))
   (NP-SBJ (PRP$ its)
       (NN balance) (NN sheet) ))
```

Figure 1: Fragment of sentence from Treebank

### Surrounding categories

The next feature added is the categories of the previous branch of the tree, and the next branch. So in the example in Figure 1, the previous category of the elliptical verb is ADVP-PRD-TPC-2, and the next category NP-SBJ. The results of using this feature are seen in Table 5, giving a 1.6 - 3.5% boost.

| Algorithm | Recall | Precision | F1 |
|---|---|---|---|
| MBL | 58.82 | 69.23 | 63.60 |
| GIS-MaxEnt | 45.09 | 81.17 | 57.98 |
| L-BFGS-MaxEnt | 64.70 | 77.95 | 70.71 |

Table 5: Effects of using the surrounding categories

### Auxiliary-final VP

For auxiliary verbs parsed as verb phrases (VP), this feature checks if the final element in the VP is an auxiliary or negation. If so, no main verb can be present, as a main verb cannot be followed by an auxiliary or negation. This feature was used by Hardt (1993) and gives a 3.4 - 6% boost to performance (Table 6).

| Algorithm | Recall | Precision | F1 |
|---|---|---|---|
| Auxiliary-final VP | 72.54 | 35.23 | 47.43 |
| MBL | 63.39 | 71.32 | 67.12 |
| GIS-MaxEnt | 54.90 | 77.06 | 64.12 |
| L-BFGS-MaxEnt | 71.89 | 76.38 | 74.07 |

Table 6: Effects of using the Auxiliary-final VP feature

### Empty VP

Hardt (1997) uses a simple pattern check to search for empty VP's identified by the Treebank, (VP (-NONE- *?*)), which is to say a

```
...
(VP (VBZ resembles)
 (NP (NNS politics) )
 (ADVP
  (ADVP (RBR more) )
  (SBAR (IN than)
   (S
    (NP-SBJ (PRP it) )
    (VP (VBZ does)
     (VP (-NONE- *?*)
      (NP (NN comedy) )))))))
```

Figure 2: Missed pseudo-gapping parse

```
...
(CC or)
(VP (VB put)
 (NP (-NONE- *-4) )
 (PP-PUT (IN into)
  (NP (NN syndication) )))
(, ,)
(SBAR-ADV (IN as)
 (S
  (NP-SBJ-1 (JJS most)
   (JJ American) (NNS programs) )
  (VP (VBP are)
   (VP (-NONE- *?*)
    (NP (-NONE- *-1) )))))))))
```

Figure 3: Missed VPE parse

VP which consists only of an empty element. This achieves 60% F1 on our development set. Our findings are in line with Hardt's, who reports 48% F1, with the difference being due to the different sections of the Treebank used.

It was observed that this search may be too restrictive to catch pseudo-gapping (Figure 2) and some examples of VPE in the corpus (Figure 3). We modify the search pattern to be '(VP (-NONE- *?*) ... )', which is a VP that contains an empty element, but can contain other categories after it as well. This improves the feature itself by 10% in F1 and gives 10 - 14% improvement to the algorithms (Table 7).

| Algorithm | Recall | Precision | F1 |
|---|---|---|---|
| Empty VP | 54.90 | 97.67 | 70.29 |
| MBL | 77.12 | 77.63 | 77.37 |
| GIS-MaxEnt | 69.93 | 88.42 | 78.10 |
| L-BFGS-MaxEnt | 83.00 | 88.81 | 85.81 |

Table 7: Effects of using the improved Empty VP feature

### Empty categories

Finally, empty category information is included completely, such that empty categories are treated as words, or leaves of the parse tree, and included in the context. Table 8 shows that adding this information results in 2.5 - 4.9% increase in F1.

| Algorithm | Recall | Precision | F1 |
|---|---|---|---|
| MBL | 83.00 | 79.87 | 81.41 |
| GIS-MaxEnt | 76.47 | 90.69 | 82.97 |
| L-BFGS-MaxEnt | 86.27 | 90.41 | 88.29 |

Table 8: Effects of using the empty categories

### Cross-validation

We perform cross-validation with and without the features developed to measure the improvement obtained through their use. The cross-validation results show a different ranking of the algorithms by performance than on the development set (Table 9), but consistent with the results for the BNC corpus. MBL shows consistent performance, L-BFGS-MaxEnt gets somewhat lower results and GIS-MaxEnt much lower. These results indicate that the confidence threshold settings of the maximum entropy models were over-optimized for the development data, and perhaps the smoothing for L-BFGS-MaxEnt was as well. MBL which was used as-is does not suffer these performance drops. The increase in F1 achieved by adding the features is similar for all algorithms, ranging from 17.9 to 19.8%.

## 5 Experiments with Automatically Parsed data

The next set of experiments use the BNC and Treebank, but strip POS and parse information, and parse them automatically using two different parsers. This enables us to test what kind of performance is possible for real-world applications.

### 5.1 Parsers used

Charniak's parser (2000) is a combination probabilistic context free grammar and maximum entropy parser. It is trained on the Penn Treebank, and achieves a 90.1% recall and precision average for sentences of 40 words or less. While Charniak's parser does not generate empty category information, Johnson (2002) has developed an algorithm that extracts patterns from the Treebank which can be used to insert empty

| | Words + POS | | | + features | | |
|---|---|---|---|---|---|---|
| Algorithm | Recall | Precision | F1 | Recall | Precision | F1 |
| MBL | 56.05 | 63.02 | 59.33 | 78.32 | 79.39 | 78.85 |
| GIS-MaxEnt | 40.00 | 57.03 | 47.02 | 65.06 | 68.64 | 66.80 |
| L-BFGS-MaxEnt | 60.08 | 70.77 | 64.99 | 78.92 | 87.27 | 82.88 |

Table 9: Cross-validation on the Treebank

categories into the parser's output. This program will be used in conjunction with Charniak's parser.

Robust Accurate Statistical Parsing (RASP) (Briscoe and Carroll, 2002) uses a combination of statistical techniques and a hand-crafted grammar. RASP is trained on a range of corpora, and uses a more complex tagging system (CLAWS-2), like that of the BNC. This parser, on our data, generated full parses for 70% of the sentences, partial parses for 28%, while 2% were not parsed, returning POS tags only.

## 5.2 Reparsing the Treebank

The results of experiments using the two parsers (Table 10) show generally similar performance. Preiss (2003) shows that for the task of anaphora resolution, these two parsers produce very similar results, which is consistent with our findings. Compared to results on the original treebank with similar data (Table 6), the results are low, which is not surprising, given the errors introduced by the parsing process. It is noticeable that the addition of features has less effect; 0-6%.

The auxiliary-final VP feature (Table 11), which is determined by parse structure, is only half as successful for RASP. Conversely, the heuristic baseline, which relies on POS tags, is more successful for RASP as it has a more detailed tagset. The empty VP feature retains a high precision of over 80%, but its recall drops by 50% to 20%, suggesting that the empty-category insertion algorithm is sensitive to parsing errors.

| | Feature | Rec | Prec | F1 |
|---|---|---|---|---|
| Charniak | close-to-punct | 34.00 | 2.47 | 4.61 |
| | heur. baseline | 45.33 | 25.27 | 32.45 |
| | aux-final VP | 51.33 | 36.66 | 42.77 |
| | empty VP | 20.00 | 83.33 | 32.25 |
| RASP | close-to-punct | 71.05 | 2.67 | 5.16 |
| | heur. baseline | 74.34 | 28.25 | 40.94 |
| | aux-final VP | 22.36 | 25.18 | 23.69 |

Table 11: Performance of features on re-parsed Treebank data

Cross-validation results (Table 12) are consistent with experiments on the development set. This is due to the fact that settings were not optimized for the development set, but left as they were from previous experiments. Results here show better performance for RASP overall.

## 5.3 Parsing the BNC

Experiments using parsed versions of the BNC corpora (Tables 13, 15) show similar results to the original results (Table 1) but the features generate only a 3% improvement, suggesting that many of the cases in the test set can be identified using similar contexts in the training data and the features do not add extra information. The performance of the features (Table 14) remain similar to those for the reparsed treebank experiments, except for empty VP, where there is a 7% drop in F1, due to Charniak's parser being trained on the Treebank only.

| | Feature | Rec | Prec | F1 |
|---|---|---|---|---|
| Charniak | close-to-punct | 48.00 | 5.52 | 9.90 |
| | heur. baseline | 44.00 | 34.50 | 38.68 |
| | aux-final VP | 53.00 | 42.91 | 47.42 |
| | empty VP | 15.50 | 62.00 | 24.80 |
| RASP | close-to-punct | 55.32 | 4.06 | 7.57 |
| | heur. baseline | 84.77 | 35.15 | 49.70 |
| | aux-final VP | 16.24 | 28.57 | 20.71 |

Table 14: Performance of features on parsed BNC data

## 5.4 Combining BNC and Treebank data

Combining the re-parsed BNC and Treebank data gives a more robust training set of 1167 VPE's and a development set of 350 VPE's. The results (Tables 16, 17) show only a 2-3% improvement when the features are added. Again, simple contextual information is successful in correctly identifying most of the VPE's.

It is also seen that the increase in data size is not matched by a large increase in performance. This may be because simple cases are already handled, and for more complex cases the context size limits the usefulness of added data. The differences between the two corpora

| | | MBL | | | GIS-MaxEnt | | | L-BFGS-MaxEnt | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Rec | Prec | F1 | Rec | Prec | F1 | Rec | Prec | F1 |
| Charniak | Words + POS | 54.00 | 62.30 | 57.85 | 38.66 | 79.45 | 52.01 | 56.66 | 71.42 | 63.19 |
| | + features | 62.66 | 71.21 | 66.66 | 48.00 | 70.58 | 57.14 | 64.66 | 72.93 | 68.55 |
| RASP | Words + POS | 55.92 | 66.92 | 60.93 | 43.42 | 56.89 | 49.25 | 51.63 | 79.00 | 62.45 |
| | + features | 57.23 | 71.31 | 63.50 | 61.84 | 72.30 | 66.66 | 62.74 | 73.84 | 67.84 |

Table 10: Results on re-parsed data from the Treebank

| | | Charniak | | | RASP | |
|---|---|---|---|---|---|---|
| Algorithm | Recall | Precision | F1 | Recall | Precision | F1 |
| MBL | 58.76 | 63.35 | 60.97 | 61.97 | 71.50 | 66.39 |
| GIS-MaxEnt | 46.22 | 71.66 | 56.19 | 56.58 | 72.27 | 63.47 |
| L-BFGS-MaxEnt | 63.14 | 71.82 | 67.20 | 64.52 | 69.85 | 67.08 |

Table 12: Cross-validation on re-parsed Treebank

| | | MBL | | | GIS-MaxEnt | | | L-BFGS-MaxEnt | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Rec | Prec | F1 | Rec | Prec | F1 | Rec | Prec | F1 |
| Charniak | Words + POS | 66.50 | 63.63 | 65.03 | 55.00 | 75.86 | 63.76 | 71.00 | 70.64 | 70.82 |
| | + features | 69.00 | 65.40 | 67.15 | 64.00 | 72.72 | 68.08 | 74.00 | 68.83 | 71.32 |
| RASP | Words + POS | 61.92 | 63.21 | 62.56 | 64.46 | 54.04 | 58.79 | 65.34 | 70.96 | 68.04 |
| | + features | 71.06 | 73.29 | 72.16 | 73.09 | 61.01 | 66.51 | 70.29 | 67.29 | 68.76 |

Table 13: Results on parsed data from the BNC

| | | Charniak | | | RASP | |
|---|---|---|---|---|---|---|
| Algorithm | Recall | Precision | F1 | Recall | Precision | F1 |
| MBL | 68.46 | 66.94 | 67.69 | 69.26 | 73.06 | 71.11 |
| GIS-MaxEnt | 61.75 | 72.63 | 66.75 | 67.49 | 72.37 | 69.84 |
| L-BFGS-MaxEnt | 71.10 | 71.96 | 71.53 | 70.68 | 72.22 | 71.44 |

Table 15: Cross-validation on parsed BNC

| | | MBL | | | GIS-MaxEnt | | | L-BFGS-MaxEnt | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Rec | Prec | F1 | Rec | Prec | F1 | Rec | Prec | F1 |
| Charniak | Words + POS | 62.28 | 69.20 | 65.56 | 54.28 | 77.86 | 63.97 | 65.14 | 69.30 | 67.15 |
| | + features | 65.71 | 71.87 | 68.65 | 63.71 | 72.40 | 67.78 | 70.85 | 69.85 | 70.35 |
| RASP | Words + POS | 63.61 | 67.47 | 65.48 | 59.31 | 55.94 | 57.37 | 57.46 | 71.83 | 63.84 |
| | + features | 68.48 | 69.88 | 69.17 | 67.61 | 71.47 | 69.48 | 70.14 | 72.17 | 71.14 |

Table 16: Results on parsed data using the combined dataset

| | | Charniak | | | RASP | |
|---|---|---|---|---|---|---|
| Algorithm | Recall | Precision | F1 | Recall | Precision | F1 |
| MBL | 66.37 | 68.57 | 67.45 | 68.21 | 73.62 | 70.81 |
| GIS-MaxEnt | 61.43 | 74.53 | 67.35 | 66.22 | 72.46 | 69.20 |
| L-BFGS-MaxEnt | 69.78 | 71.65 | 70.70 | 71.00 | 73.22 | 72.09 |

Table 17: Cross-validation on combined dataset

may also limit the relevance of examples from one to the other.

## 6 Summary and Future work

This paper has presented a robust system for VPE detection. The data is automatically tagged and parsed, syntactic features are extracted and machine learning is used to classify instances. This work offers clear improvement over previous work, and is the first to handle un-annotated free text, where VPE detection can be done with limited loss of performance compared to annotated data.

- Three different machine learning algorithms, Memory Based Learning, GIS-based and L-BFGS-based maximum entropy modeling are used. They give similar results, with L-BFGS-MaxEnt generally giving the highest performance.

- Two different parsers were used, Charniak's parser and RASP, achieving similar results in experiments, with RASP results being slightly higher. RASP generates more fine-grained POS info, while Charniak's parser generates more reliable parse structures for identifying auxiliary-final VP's.
- Experiments on the Treebank give 82% F1, with the most informative feature, empty VP's, giving 70% F1.
- Re-parsing the Treebank gives 67% F1 for both parsers. Charniak's parser combined with Johnson's algorithm generates the empty VP feature with 32% F1.
- Repeating the experiments by parsing parts of the BNC gives 71% F1, with the empty VP feature further reduced to 25% F1. Combining the datasets, final results of 71-2% F1 are obtained.

Further work can be done on extracting grammatical relation information (Lappin et al., 1989; Cahill et al., 2002), or using those provided by RASP, to produce more complicated features. While the experiments suggest a performance barrier around 70%, it may be worthwhile to investigate the performance increases possible through the use of larger training sets. In the next stage of work, we will use machine learning methods for the task of finding antecedents. We will also perform a classification of the cases to determine what percentage can be dealt with using syntactic reconstruction, and how often more complicated approaches are required.

## References

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565.

E. Briscoe and J. Carroll. 2002. Robust accurate statistical annotation of general text. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, Las Palmas, Gran Canaria.

Aoife Cahill, Mairead McCarthy, Josef van Genabith, and Andy Way. 2002. Evaluating automatic f-structure annotation for the penn-ii treebank. In *Proceedings of the First Workshop on Treebanks and Linguistic Theories (TLT 2002)*, pages 42–60.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Meeting of the North American Chapter of the ACL*, pages 132–139.

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2002. Tilburg memory based learner, version 4.3, reference guide. Downloadable from http://ilk.kub.nl/downloads/pub/papers/ilk0210.ps.gz.

Mary Dalrymple, Stuart M. Shieber, and Fernando Pereira. 1991. Ellipsis and higher-order unification. *Linguistics and Philosophy*, 14:399–452.

Robert Fiengo and Robert May. 1994. *Indices and Identity.* MIT Press, Cambridge, MA.

Daniel Hardt. 1993. *VP Ellipsis: Form, Meaning, and Processing.* Ph.D. thesis, University of Pennsylvania.

Daniel Hardt. 1997. An empirical approach to vp ellipsis. *Computational Linguistics*, 23(4).

Mark Johnson. 2002. A simple pattern-matching algorithm for recovering empty nodes and their antecedents. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics.*

Andrew Kehler and Gregory Ward. 1999. On the semantics and pragmatics of 'identifier so'. In Ken Turner, editor, *The Semantics/Pragmatics Interface from Different Points of View (Current Research in the Semantics/Pragmatics Interface Series, Volume I).* Amsterdam: Elsevier.

Andrew Kehler. 1993. A discourse copying algorithm for ellipsis and anaphora resolution. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics (EACL-93)*, Utrecht, the Netherlands.

Torbjorn Lager. 1999. The mu-tbl system: Logic programming tools for transformation-based learning. In *Third International Workshop on Computational Natural Language Learning (CoNLL'99).* Downloadable from http://www.ling.gu.se/ lager/mutbl.html.

Shalom Lappin, I. Golan, and M. Rimon. 1989. Computing grammatical functions from configurational parse trees. Technical Report 88.268, IBM Science and Technology and Scientific Center, Haifa, June.

Shalom Lappin. 1993. The syntactic basis of ellipsis resolution. In S. Berman and A. Hestvik, editors, *Proceedings of the Stuttgart Ellipsis Workshop, Arbeitspapiere des Sonderforschungsbereichs 340, Bericht Nr. 29-1992.* University of Stuttgart, Stuttgart.

Shalom Lappin. 1996. The interpretation of ellipsis. In Shalom Lappin, editor, *The Handbook of Contemporary Semantic Theory*, pages 145–175. Oxford: Blackwell.

G. Leech. 1992. 100 million words of english : The British National Corpus. *Language Research*, 28(1):1–13.

Robert Malouf. 2002. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the Sixth Conference on Natural Language Learning (CoNLL-2002)*, pages 49–55.

M. Marcus, G. Kim, M. Marcinkiewicz, R. MacIntyre, M. Bies, M. Ferguson, K. Katz, and B. Schasberger. 1994. The Penn Treebank: Annotating predicate argument structure. In *Proceedings of the Human Language Technology Workshop.* Morgan Kaufmann, San Francisco.

Leif Arda Nielsen. 2003a. A corpus-based study of verb phrase ellipsis. In *Proceedings of the 6th Annual CLUK Research Colloquium*, pages 109–115.

Leif Arda Nielsen. 2003b. Using machine learning techniques for VPE detection. In *Proceedings of RANLP*, pages 339–346.

Judita Preiss. 2003. Choosing a parser for anaphora resolution. In *Proceedings of DAARC*, pages 175–180.

R. Quinlan. 1993. *C4.5: Programs for Machine Learning.* San Mateo, CA: Morgan Kaufmann.

Adwait Ratnaparkhi. 1998. *Maximum Entropy Models for Natural Language Ambiguity Resolution.* Ph.D. thesis, University of Pennsylvania.

Stuart Shieber, Fernando Pereira, and Mary Dalrymple. 1996. Interactions of scope and ellipsis. *Linguistics and Philosophy*, 19(5):527–552.