A corpus-based study of Verb Phrase Ellipsis Identification and Resolution

Leif Arda Nielsen

Thesis submitted to the University of London for the degree of Doctor of Philosophy

Department of Computer Science
King's College London
2005

Abstract

Although considerable work exists on the subject of ellipsis resolution, there has been very little empirical, corpus-based work on it. I propose a system which will take free text and (i) detect instances of Verb Phrase Ellipsis, (ii) identify their antecedents and (iii) resolve the ellipses, providing the components for an end-to-end solution.

The central claim of this thesis is that it is possible to implement each of these stages using knowledge-poor approaches while achieving high coverage. To demonstrate this, robust and accurate methods have been developed, using machine learning techniques where applicable. Each stage is tested on corpus data, giving significant improvements over previous work and providing new insights.

The results obtained confirm the claims made. While the number of cases which are problematic for the approaches developed are not insignificant, it is shown that the majority of cases can be handled with success. These results hold for automatically parsed data as well as manually coded ones, allowing for a robust system that can be used for real-world applications.

Acknowledgements

I am greatly indebted to my supervisor, Shalom Lappin, who has been encouraging, helpful, and nothing less than all that can be expected of a supervisor throughout. I could not have wished for a more knowledgeable, or nicer, person to work with.

I am also indebted to Daniel Hardt, who has given me much help, and whose work I have been inspired by. Thanks too to Jonathan Ginzburg for his kind and helpful advice.

I would like to offer my gratitude to my examiners, Rob Gaizauskas and Pat Healey, for their suggestions and criticism, which has much improved this thesis.

I am thankful to my family, who have supported my desire to study, and have accepted having an absent son.

The person who deserves the greatest thanks, however, is Naila Mimouni. Without her constant friendship, motivation, cheerfulness and support, I couldn't have pulled through with this. That she has put up with all of my problems during this time is a testament to what a treasure of a friend she is, and to her generosity and character.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Ellipsis is an anaphoric process where a syntactic constituent is left unexpressed, but can be reconstructed from an antecedent in the context. Some forms of ellipsis are seen in Example (1).

(1) a. John read the paper before Bill did.

b. John read Sam's story, and Tom read Bill's.

c. John asks that we go to the meeting, and Harry wants to know when.

d. John writes plays, and Bill does novels.

e. Sam teaches in London, and Lucy in Cambridge.

f. Bill wrote reviews for the journal last year, and articles this year.

In (1a) a verb phrase is missing (VP ellipsis, VPE), in (1b) a noun (N ellipsis), and in (1c) an inflectional phrase (sluicing). Example (1d) is an instance of pseudo-gapping, (1e) of gapping, and (1f) of bare ellipsis.

Ellipsis is a linguistic phenomenon that has received considerable attention, mostly focusing on its interpretation. Insight has been gained through work aimed at discerning the procedures and the level of language processing at which ellipsis resolution takes place. Such work has generally resulted in two views: syntactic and semantic. While the syntactic account (Sag, 1976; Fiengo and May, 1994;

Gregory and Lappin, 1997; Kennedy and Merchant, 2000) suggests that ellipsis resolution involves copying syntactic material from the antecedent clause to the ellipsis site, the semantic account (Dalrymple et al., 1991; Kehler, 1993; Shieber et al., 1996) argues that this material is obtained from semantic representations.

1.1 Aim of this thesis

For many natural language processing applications, such as machine translation and information extraction, resolving ellipses prior to further processing may be desirable, and perhaps necessary for perfect results. Both the syntactic and semantic accounts of ellipsis have their strengths and weaknesses, but they have so far not been validated using a corpus-based, empirical approach, meaning that their actual performance is unknown. Furthermore, while these approaches take difficult cases into account, they do not deal with noisy or missing input, which are unavoidable in NLP applications. They also do not allow for focusing on specific domains or applications. It therefore becomes clear that a robust, trainable approach to ellipsis resolution is needed.

Several steps of work are necessary for ellipsis resolution :

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

1. **Detecting ellipsis occurrences** First, elided verbs need to be found.
2. **Identifying antecedents** The correct verb phrase corresponding to the antecedent of the VPE needs to be selected from the surrounding sentences.
3. **Resolving the ellipses** For most cases of ellipsis, copying of the antecedent clause to the VPE site, with or without some simple transformations, is enough for resolution. For other cases, for example when ambiguity exists, a method for detection is necessary so that they can be handled by other modules.

Existing methods usually do not deal with the detection of elliptical sentences or the identification of the antecedent and elided clauses within them, but take

them as given, concentrating instead on the resolution of ambiguous or difficult cases.

The claim of this thesis is that it is possible to produce the components of a VPE resolution system, using a corpus-based, knowledge-poor approach[1]. The detection of ellipsis occurrences and the identification of their antecedents will be the main points of focus. Adopting an empirical, corpus-based approach will enable us to quantitatively assess performance. Finally, a classification of the data will be performed depending on the complexity of resolution necessary. A simple, rule-based system will be built to resolve the majority of cases in the data.

I will investigate the use of machine learning (ML) techniques for the first two stages of ellipsis resolution. The use of machine learning algorithms allows the production of robust systems that require little expert knowledge, and also defines the work in a way that can be adapted to similar domains with minimal effort. For the last stage generating rules by hand is simple, and will be used instead of ML techniques.

The data used will be written English texts, from genres that range from articles to plays. I have chosen to concentrate on VP ellipsis, as an informal look at the data seen while collecting the first hundred VPEs suggests that it is far more common than other forms of ellipsis. Pseudo-gapping, an example of which is seen in Example (1d), 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), and neither are certain comparatives or inversions (see section 2.1.4).

1.2 Overview

Chapter 2 aims to provide the context for this work, and is divided into two parts. In section 2.1 previous work and the theory associated with VPE is de-

[1] While the system proposed would contain the steps necessary for VPE resolution, post-processing might be necessary before its output could be used in other NLP systems. Tasks such as co-indexing NPs, pronominals and traces in the resolved sentence would have to be left for other modules.

scribed. In section 2.2 the choice of using ML approaches will be justified and its use in NLP summarized. The basic principles, advantages and disadvantages of the particular ML algorithms used will also be explained.

Chapter 3 describes how the experiments were performed, the data used in the experiments, and the criteria of success.

Chapter 4 deals with experiments for detecting elliptical verbs. Experiments with increasing amounts of informational complexity are performed.

Chapter 5 describes experiments on finding the antecedents for the VPEs detected in the previous step. A baseline is built using a set of heuristics, with further experiments utilizing these in ML.

Chapter 6 provides an analysis and classification of the data found in the corpus in terms of resolution complexity. A simple rule-based system is built which resolves the less complex cases in the data.

Chapter 7 provides a summary and conclusion.

Chapter 2

Background

For the work done in this thesis, linguistic insights are coupled with machine learning approaches. This necessitates a review of both areas, which are presented in this chapter.

2.1 Verb Phrase Ellipsis

The resolution of ellipsis in general, and VP ellipsis in particular, is a topic on which a large amount of research has been focused. The size of the literature generated as a result of this research prohibits an exhaustive review of the area, or a complete enumeration of the methods proposed. Instead, this chapter will summarize the facts concerning the major approaches to have emerged from the area, and a number of specific implementations as well.

VPE is signalled by an auxiliary, semi-auxiliary or modal[1] verb stranded without a VP (Examples (3a), (3b)). To interpret the elided VP, its antecedent clause needs to be identified, which is usually a trivial task for humans. In cases such as (3a), where the antecedent clause contains a pronoun, ambiguity exists. One reading is that Bill loves John's wife, which is termed the *strict* reading. The other reading is that Bill loves his own wife, termed the *sloppy* reading. Contextual information is needed to perform a choice between the readings.

[1]These three verb types combined will be referred to as auxiliary for brevity.

(3) a. John loves his wife. Bill does too. [love his wife]

b. But although he was terse, he didn't rage at me the way I expected him to. [rage at me]

Considerable work has gone into determining how VPE is resolved, with two main approaches emerging, semantic and syntactic, outlined in sections 2.1.1 and 2.1.2, respectively. Afterwards, an attempt to combine aspects of these two approaches will be discussed (section 2.1.3).

2.1.1 Semantic approach

Semantic views of ellipsis (Dalrymple et al., 1991; Shieber et al., 1996; Hardt, 1999) argue that ellipsis is resolved at a semantic level of representation, using generalized mechanisms for the recovery of meanings from context. The prototypical example of such analyses, the equational, higher-order unification-based analysis introduced by Dalrymple et al. (1991) asserts that the antecedent-clause provides information which is matched with the elided clause to extract a property which applies to both, which is then used to resolve the ellipsis. This analysis is seen as it applies to (3a).

The semantic representation of the antecedent clause is derived as :

(4) love(John, wife-of(John))

Where John is a *primary occurrence*, which must be abstracted over when generating a solution at the ellipsis site[2]. From this a property P can be derived that will generate the reading at the ellipsis site :

[2]If primary occurrences were included in solutions, and the first two items in (7) were retained, the list of available readings in (8) would include :

(5) a. λx.love(John, wife-of(John))(Bill) = love(John, wife-of(John))

b. λx.love(John, wife-of(x))(Bill) = love(John, wife-of(Bill))

Where the first reading is that John loves his own wife, and the second that John loves Bill's wife. This constraint enforces the parallelism in the ellipses, and prevents such overgeneration.

(6) P(John) = love(John, wife-of(John))

A suitable value for P is needed to solve equation (6) :

(7) a. P $\mapsto \lambda$ x.love(John, wife-of(John))

b. P $\mapsto \lambda$ x.love(John, wife-of(x))

c. P $\mapsto \lambda$ x.love(x, wife-of(John))

d. P $\mapsto \lambda$ x.love(x, wife-of(x))

Ignoring (7a) and (7b) as they contain John, the sentence is interpreted as :

(8) a. λx.love(x, wife-of(John))(Bill) = love(Bill, wife-of(John))

b. λx.love(x, wife-of(x))(Bill) = love(Bill, wife-of(Bill))

The first reading is the strict one, where Bill loves John's wife, and the second the sloppy reading, where Bill loves his own wife. While some approaches derive the property from the meaning of the entire source clause (Dalrymple et al., 1991; Kehler, 1993), others derive it from the meaning of a VP (Hardt, 1992b; Hardt, 1999).

Examples such as (9), (10), (11) and (12) where the syntactic structure necessary for resolution is not readily available, are usually argued as proving the need for semantic analyses, as the antecedents cannot be straightforwardly derived from the syntactic representations

(9) and (12) are cases where there is a voice mismatch, and (11) has a split antecedent.

(9) A lot of this material can be presented in a fairly informal and accessible fashion, and often I do. [present a lot of this material in a fairly informal and accessible fashion][3]

[3]From (Chomsky, 1982)

(10) China is a country that Joe wants to visit, and he will too, if he gets enough money.[4]

(11) Mary wants to go to Spain and Fred wants to go to Peru, but because of limited resources, only one of them can. [go to Spain or go to Peru][5]

(12) Avoid getting shampoo in eyes - if it does, flush thoroughly with water. [get in eyes][6]

It has also been suggested (Dalrymple et al., 1991; Hardt, 1993; Kehler, 2002b) that cases such as (14), (15) and (16) also strengthen the case for a semantic approach, where (14) and (16) contain nominalized antecedents[7].

(14) Harry used to be a great speaker, but he can't any more, because he lost his voice. [speak][8]

(15) Mary and Irv want to go out, but Mary can't, because her father disapproves of Irv. [go out with Irv][9]

[4]From (Lappin, 1996)

[5]From (Webber, 1979)

[6]From (Dalrymple et al., 1991)

[7]Lappin (1996) however, argues that this is not necessarily the case, as the semantic representations for the correct antecedents are not derived from the sentences themselves, but from structures such as *'Mary wants to go out with Irv'*, *'Harry used to speak'*, and *'The ancients were wise'*, which need to be derived by use of inference. Indeed, in the last example the elided sentence is followed by one where the ellipsis is expanded, suggesting that the author is aware of the complexity of the antecedent, which combines nominalization, an unspoken verb 'be', and an indexical style. Another interesting example is :

(13) a. Bob's mother cleans up after him all the time.

b. I'm surprised; most parents these days won't. [clean up after their children] (From (Kehler, 2002a))

Here the material for an antecedent is no more available to a semantic analysis than it is to a syntactic one, without the use of inference. An inference module that deals with such cases could be interfaced with either a semantic or syntactic resolution system equally well, and once inference is involved both approaches can deal with the problematic sentences. This suggests that these examples do not contribute to the discussion on the level of ellipsis resolution one way or the other.

[8]From (Hardt, 1993)

[9]From (Webber, 1979)

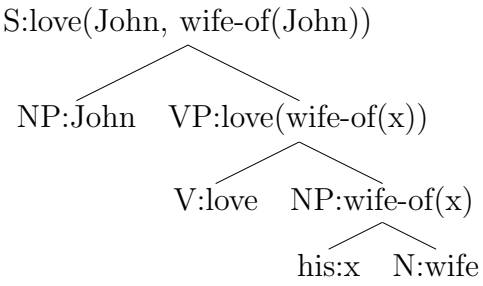

Figure 2.1: Syntax and Semantics for *John loves his wife*

(16) Ancients, wisdom of the. They were not. [wise]
They were not wise.[10]

2.1.2 Syntactic approach

The syntactic account holds that elided material contains syntactic structure, but there are two subdivisions as to how VPE occurs. Under the reconstruction approach (Wasow, 1972; Williams, 1977; Haik, 1987; Lappin and McCord, 1990; Kitagawa, 1991; Lappin, 1993; Fiengo and May, 1994; Hestvik, 1995; Lappin, 1996), the data at the ellipsis site is recovered from the antecedent structure, and the semantics of the reconstructed clause is the same as the antecedent, as they share the same syntactic structure. Under the deletion approach (Sag, 1976; Hankamer, 1979; Tancredi, 1992; Kennedy and Merchant, 2000), VPE occurs when existing syntactic material is not uttered under suitable conditions. Syntactic approaches also diverge as to which level VPE operates on; surface syntax (Lappin, 1993; Lappin, 1996) or some level of syntactic logical form (Kitagawa, 1991; Fiengo and May, 1994). Using the reconstruction approach, (3a) is resolved in the steps seen in Figures 2.1, 2.2 and 2.3.

The strength of the syntactic account lies in its ability to predict unacceptable cases using syntactic constraints.

(17) * John$_i$ defended himself$_i$, and Bob$_j$ did too. [defend himself$_i$][11][12]

[10] Guardian Weekend 23.10.2004, "You don't know what you've got 'til it's gone"
[11] Originally from (Kitagawa, 1991).
[12] The convention of using a * in front of an example to signify its unacceptability, and a ? to signify that its acceptability is questionable will be adopted.

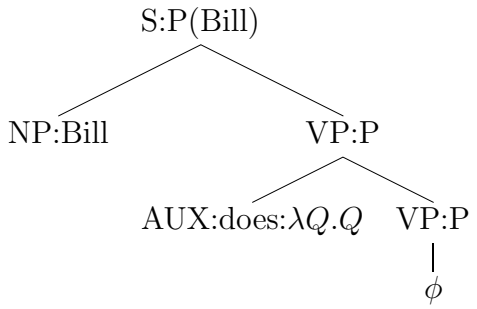

Figure 2.2: Syntax and Semantics for *Bill does [too]*

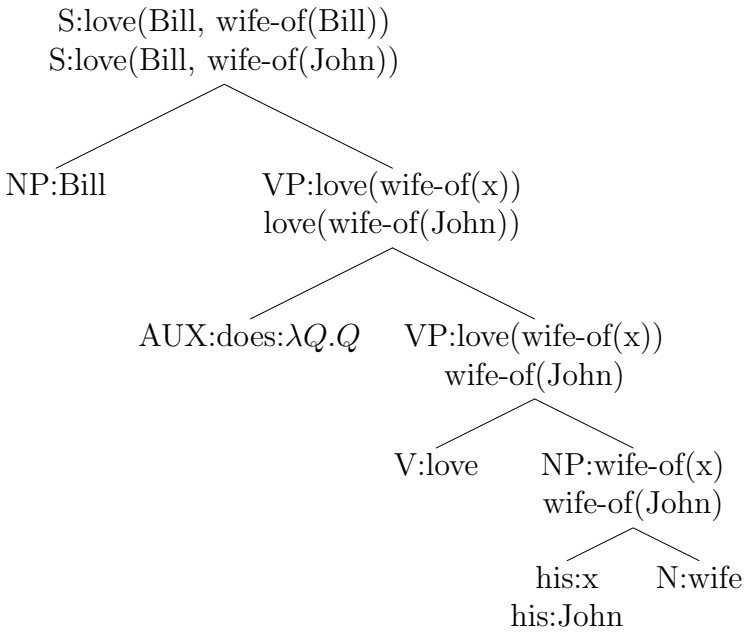

Figure 2.3: Syntax and Semantics for *Bill does [too]*, reconstructed

(17), where the strict reading is presented, is correctly predicted as being unacceptable due to syntactic criteria, and only the sloppy reading is allowed. This is due to Condition A[13], which states that an anaphor must be bound in its bind-

[13]Binding theory (Chomsky, 1981) assigns interpretations for three types of noun phrases :

- Reflexives and reciprocals (anaphors) : himself, herself, ourselves etc.
- Non-reflexive pronouns (pronominals) : he, she, his, your etc.
- Full NPs and names (r-expressions) : John, this, the Judge etc.

It stipulates three constraints on them :

Condition A A reflexive pronoun (anaphor) must be bound in its binding domain, i.e. it must have an antecedent within its local clause.

ing domain. The reconstructed reflexive can only be interpreted sloppily, due to the need for a locally-bound antecedent. Semantic theories do not predict the unacceptability of this example.

Condition C predicts the unacceptability of (18)[14], as it would require the pronoun *he* to bind its co-referring full NP *Bill*.

(18) * The lawyer defended Bill$_i$, and he$_i$ did too. [defend Bill$_i$]

Lappin (1996) cites Examples (19,20)[15], which illustrate subjacency effects within antecedent contained ellipsis sites. This again is a syntactic constraint that is not available to semantic analyses.

(19) a. John read everything which Mary believes that he did.

b. * John read everything which Mary believes the claim that he did.

c. * John read everything which Mary wonders why he did.

(20) a. This is the book which Max read before knowing that Lucy did.

b. * This is the book which Max read before hearing the claim that Lucy did.

c. ? This is the book which Max read before knowing why Lucy did.

The examples discussed cannot be detected as being unacceptable on a purely semantic approach, and show the appeal of the syntactic account.

Furthermore, some cases given as supporting the semantic approach can be handled within syntactic reconstruction. Fiengo and May (1994) suggest *vehicle*

Condition B A non-reflexive pronoun (pronominal) must be free in its binding domain, i.e. it must not have an antecedent within its local clause.

Condition C A full NP (r-expression) must not be bound, i.e. must be free.

[14]Originally from (Lappin, 1993).
[15]Originally from (Haik, 1987) and (Fiengo and May, 1994).

change as a way of dealing with such cases where there is a discrepancy between the antecedent form and the ellipsis site. Vehicle change addresses this by allowing the syntactic form of an argument to be changed during reconstruction.

In (21), copying all of the antecedent results in syntactic problems. Vehicle change allows for the name to be changed to a trace, which gives the correct reconstruction.

(21) a. Dulles suspected Philby, who Angleton did, too.[16]

b. * Dulles suspected Philby, who Angleton suspected Philby, too.

c. Dulles suspected Philby$_1$, who Angleton suspected t$_1$, too.

In (22), straightforward copying of the antecedent would result in a Condition C violation. Allowing the name to be changed to a pronoun, however, results in a correct reconstruction.

(22) a. Mary loves John$_1$, and he$_1$ thinks that Sally does, too.[17]

b. * Mary loves John$_1$, and he$_1$ thinks that Sally loves John$_1$, too.

c. Mary loves John$_1$, and he$_1$ thinks that Sally loves him$_1$, too.

Returning to (10), it is seen that this can be handled by vehicle change as well by allowing traces to be reconstructed as pronouns :

(23) a. China is a country that Joe wants to visit, and he will too, if he gets enough money.

b. China is a country that Joe wants to visit t$_1$, and he will visit it$_1$ too, if he gets enough money.

[16](Fiengo and May, 1994), p.219
[17](Fiengo and May, 1994), p.220

2.1.3 A composite model

Kehler (1995; 2002b) offers a mixture of the syntactic and semantic approaches, and asserts that coherence relations determine when each approach is applicable.

Kehler distinguishes between three broad types of connections between utterances that can be used to form a coherent discourse. In (24a) a Cause-Effect relation is operative, as the second utterance must be taken to explain the first for the passage to be coherent. In (24b) a Resemblance relation is operative. Without things in common between Bill and George, the passage lacks coherence, but if it were given that George refers to George Bush and Bill to Bill Clinton, the common topic results in greater coherence. Finally, in (24c), the Association relation is operative, as the passage is not coherent without forming the assumption that George Bush must have been on the train.

(24) a. John took a train from Paris to Istanbul. He likes spinach.

b. Bill likes to jog, and George hates broccoli.

c. At 5:00 a train arrived in Chicago. At 6:00 George Bush held a press conference.

Kehler argues that for Cause-Effect relations, the arguments are simply the propositional semantics of each utterance. For Resemblance relations, on the other hand, syntactic parallelism is required. Kehler takes the semantic approach to ellipsis resolution as the default, due to the pronominal characteristics of VPE (Chao, 1987; Hardt, 1992a). He then predicts three consequences of these assumptions :

- Cause-Effect relations will pattern with semantic resolution, as the required information is available.
- Resemblance relations will require parallelism between the antecedent clause and the VPE site.
- If there is syntactic information below the VP level that is necessary to establish parallelism, reconstruction will be necessary.

In support of the prediction on Cause-Effect relations, Examples (9), (11), (12), (14), and (15) show the need for semantic resolution. These are all examples where a syntactic account would fail, as there is no clear syntactic antecedent.

Furthermore, the existence of Cause-Effect relations can result in felicitous sentences which would be predicted to violate syntactic conditions. The three sentences in examples (25a), (25b) and (25c) show instances where Conditions A, B and C, respectively, are violated, but are still felicitous.

(25) a. Bill$_i$ defended himself$_i$ against the accusations because his lawyer$_j$ couldn't. [defend himself$_i$]

b. John$_i$'s lawyer defended him$_i$ because he$_i$ couldn't. [defend him$_i$]

c. I expected Bill$_i$ to win even when he$_i$ didn't. [expect Bill$_i$ to win]

In keeping with the prediction on Resemblance relations, examples (17) and (18) show the need for syntactic reconstruction when Resemblance relations are present. This intuition is strengthened by a comparison of various levels of parallelism in two forms of sentences where there is a mismatch between the syntactic form of the antecedent and the VPE site.

The voice alteration in Example (9) is acceptable in a Cause-Effect relation, but in a similar Resemblance relation the interpretation becomes difficult, as seen in Example (26a). This is because the Resemblance relation requires parallelism between the VPE site and the antecedent clause, which is missing here. Even with a connective indicating a Cause-Effect relationship ('even though'), the Parallel relation still makes the reading in Example (26b) difficult. Reducing the parallelism by changing the auxiliary from 'did' to 'had', which is less parallel to 'was', as in Example (26c), however, makes the sentence acceptable.

(26) a. * A lot of this material was presented in a fairly informal and accessible fashion by John, and Bob did too. [present a lot of this material in a fairly informal and accessible fashion]

b. ? A lot of this material was presented in a fairly informal and accessible fashion by John, even though Bob did (too). [present a lot of this material in a fairly informal and accessible fashion]

c. A lot of this material was presented in a fairly informal and accessible fashion by John, even though Bob already had. [presented a lot of this material in a fairly informal and accessible fashion]

The same grades of acceptability are seen when applied to cases of nominalization, as in Example (14), with an adapted example in 27a. This is not acceptable, as the Resemblance relation is not satisfied with the level of parallelism between the clauses. Example (27b) is questionable, but more acceptable, due to the addition of 'because', which shifts the balance of the relation toward Cause-Effect. Changing the auxiliary to one that is less parallel results in a sentence which is acceptable (27c).

(27) a. * This letter provoked a response from Bush, and Clinton did too. [respond]

b. ? This letter provoked a response from Bush because Clinton did (too). [respond]

c. This letter provoked a response from Bush because Clinton already had. [responded]

2.1.4 Pseudogapping, comparatives, inversion and 'do so/it/that' anaphora

(Lappin, 1996) argues that the treatment of pseudo-gapping and VPE is identical under S-structure based syntactic reconstruction. This type of reconstruction posits that an elided VP is only partially empty and contains the trace of the *wh*-phrase, when the *wh*-phrase binds an argument in an elided VP. Under such an analysis, all the cases in (28) are similar, ranging from a full VPE to pseudo-gapping, as components are added. Given this similarity, pseudogapping will be included in our analysis.

(28) a. John sent flowers to Lucy before Max did.

b. John sent flowers to Lucy before Max did chocolates.

c. John sent flowers to Lucy before Max did to Mary.

d. John sent flowers to Lucy before Max did chocolates to Mary.

Comparatives such as Example (29a) can be seen as cases of VPE, but their resolution requires an added level of complexity not found in most instances of VPE, and will be set aside for future work. Cases where there is a comparative relation, but the antecedent is straightforward, such as Example (29b) will be retained.

(29) a. The judge says he can't discuss in detail how he will defend himself at his trial, although he contends that if he were as corrupt as state prosecutors believe, he would be far wealthier than he is. [wealthy]

b. And do you really think that the world outside Poland will care any more than we do ? [care]

Cases of inversion, such as *"Going, is she?"* are excluded from the experiments as they do not require any reconstruction, but simply reordering.

(Hankamer and Sag, 1976) present a division of anaphors into two categories: *'deep'* and *'surface'*, as illustrated in Example (30). Surface anaphors require a suitable syntactic antecedent, and include cases such as VPE and gapping. Deep anaphors can be situationally evoked, and only require a semantic referent. This includes cases such as pronominals, and *do it* and *do that* anaphora. Therefore these last two cases will not be included in our analysis.

(30) A peace agreement in the former Yugoslav republic needs to be drawn up.[18]

a. An agreement in North Korea does too. [VPE (surface)]

b. * Jimmy Carter volunteered to. [VPE (surface)]

c. Jimmy Carter volunteered to do it. [event anaphora (deep)]

[18]From (Kehler and Ward, 1999).

(Kehler and Ward, 1999; Ward and Kehler, 2002) show that the *do* in *do so* is not an auxiliary, but a main verb, as it cannot be inverted, permits *so*, and is limited to non-stative events. The *so* of *do so* is an adverb, as it doesn't passivize or cleft. They show that *do so* and *so doing* are forms of hyponymic reference, as seen in (31).

(31) John Gotti dispensed with his mob boss by shooting him in broad daylight, with plenty of witnesses around.

a. By so shooting him, Gotti established himself as his victim's likely successor. [same verb]

b. By so murdering him, Gotti established himself as his victim's likely successor. [more general hyponym]

c. By so doing, Gotti established himself as his victim's likely successor. [most general hyponym]

As a result of this, it is concluded that *do so* and *so doing* are anaphoric and do not require a matching syntactic antecedent. These constructions are used in cases where an event in the hearer's mental model is being accessed, and therefore they will not be included in our study.

2.2 Machine Learning

Machine learning (ML) (Langley, 1996; Mitchell, 1997; Manning and Schutze, 1999; Jurafsky and Martin, 2000) is the process of automatic acquisition of a computational model (or system) through exposure to experience (a corpus of events). As larger amounts of suitable data in the form of annotated corpora become available, coupled with the increase in processing power, ML approaches are becoming prevalent in NLP.

Most NLP problems can be interpreted as classification tasks, on which a large amount of experience exists in ML. ML has been successfully applied to numerous NLP tasks, such as information extraction, machine translation, grammar

learning, part of speech (POS) tagging, chunking, parsing, and word-sense disambiguation.

Machine learning approaches offer several advantages over hand-built NLP systems :

- For many tasks, such as grammar induction, generating all the rules needed can be an intractable problem, or at the very least an expensive one. ML can reduce development costs by minimizing the need for expert information.
- Additionally, rules can interact in complicated and unpredictable ways, making manual development difficult. ML approaches can deal with this due to their statistical nature, which can weigh the different factors according to their contribution to the criteria of success.
- ML systems can improve performance over time as new data are added.
- ML approaches can provide robustness not available to hand-constructed systems.
- ML can be useful for discovering patterns/rules in data that may be missed by experts.
- It is also possible to bootstrap an application using existing manually developed systems and then improve performance using ML. Therefore, ML systems can be adopted without throwing away existing resources.

Machine learning approaches can be generally divided into two types : supervised and unsupervised. In supervised learning a set of inputs is provided to the learning algorithm along with the correct results (gold standard), and learning consists of modifying the model to correctly predict new data. The creation of the labelled data needed for supervised learning can be laborious, and presents the bottleneck for most applications using supervised learning. Given the labelled data, the success of the system depends on the choice of features presented to the learner. Supervised learning approaches are generally similar to each other, are well understood, and offer similar performance. Supervised learning is the most common approach used in NLP applications.

In unsupervised learning the correct results are not provided, and the algorithm is expected to find patterns and groupings in the data. Unsupervised learning has the advantage that it does not require labelled data. Its disadvantage is that it generally performs worse than supervised learning, and on some tasks it may not work at all. The reason for this is that any dataset is likely to contain any number of patterns, but extracting the desired relations is difficult. Despite these issues, unsupervised learning has been successfully applied to tasks such as word sense disambiguation (Yarowsky, 1995), morphology induction (Goldsmith, 2001) and syntactic induction (dependency and constituency) (Klein and Manning, 2004), where it produces systems which rival supervised learning approaches.

Semi-supervised learning attempts to strike a balance between the two approaches. First, supervised learning is used on a small labelled corpus, which should point the system in the right direction. Afterwards, unsupervised learning is used with a large corpus, broadening the coverage of the system. It has been applied to tasks such as POS tagging (Merialdo, 1994).

Although unsupervised learning is interesting for its scientific implication - that language learning is possible with a generalized induction scheme and minimal language model bias - and because it may eventually help overcome the annotated data bottleneck that exists for NLP, its use in NLP is relatively new. Throughout this thesis supervised learning will be used, as it achieves higher performance.

ML comes in a large variety of approaches, and the rest of this chapter will outline those used in this thesis. One of those described (Maximum Entropy) is statistical, while the rest are symbolic, or weakly-statistical, in that they do not directly use statistics in the classification process. The algorithms also apply different inductive biases to the learning process: decision trees favour the simplest solution; memory based learning has the advantage of storing rare cases; transformation based learning accepts learning biases that are input in the form of rule templates; and maximum entropy approaches aim to incorporate no bias whatsoever. With the exception of transformation based learning, all the algorithms can produce confidence estimates for decisions, as well as binary outcomes.

There are many more ML models, many of which have also been applied to NLP tasks with good results, such as Support Vector Machines (Cristianini and Shawe-Taylor, 2000), neural nets (Haykin, 1994), and genetic algorithms (Winter

et al., 1995), to name a few. Due to space considerations I limit myself to the algorithms discussed here, which constitute a representative set of current ML approaches.

2.2.1 Transformation-based learning

Transformation-based learning (TBL) (Brill, 1993b) is an error-minimizing, greedy search[19] which compares an initial guess to the gold standard. Using the available rule templates it finds the best series of transformations required to generate the gold standard.

The learning process begins by constructing an initial guess. This guess can be as simple or complicated as desired. For a part-of-speech (POS) tagger, this can be a very simplistic guess of noun as POS for every word, because it is the most common POS tag. Or it can be as complicated as hand-annotated data from a single annotator, where the aim is to find systematic mistakes in his/her annotation.

In the next step this initial guess is compared to the gold standard. Using pre-specified transformation templates of the form "*if X and Y then replace X with Z*"[20], the algorithm searches for every instantiation of X, Y and Z to find the rule that makes the largest net change to the initial guess to bring the output of the

[19] A greedy algorithm chooses the rule with the highest payoff at any given step. This simplifies the procedure, but is likely to result in a non-optimal system. A non-greedy algorithm would require several passes over the data, adjusting the existing rules. C4.5 and SLIPPER (see following sections) also use greedy algorithms.

[20] These templates can be very simple, yet effective, such as those used by Brill to build a POS tagger (1993b, pp. 83) :

Change a tag from X to Y if :

- The previous (following) word is tagged with Z
- The previous word is tagged with Z and the next word with W
- The following (preceding) two words are tagged with Z and W
- One of the two preceding (following) words is tagged with Z
- One of the three preceding (following) words is tagged with Z
- The word two words before (after) is tagged with Z

Or they can be very complex. However, as the algorithm has to search through every instantiation of the variables, the problem can quickly become computationally very expensive.

rule closest to the gold standard. The initial guess is replaced with the data that is generated as a result of this step, and the process is repeated until no rules can be found that make substantial improvements.

This approach was developed specifically for corpus-based NLP, unlike the other ML models, which have been adopted from other ML tasks. One of the features of TBL is that it allows for linguistic biases being provided in the rule templates, but it does not rely on them.

TBL has been applied to a number of NLP tasks, such as POS tagging - supervised (Brill, 1992; Brill, 1993b; Brill, 1995) and unsupervised (Brill, 1997), parsing (Brill, 1993b; Brill, 1993a; Brill, 1996), prepositional phrase attachment (Brill and Resnik, 1994), text chunking (Ramshaw and Marcus, 1995), correcting article-noun agreement and comma placement problems in Danish (Hardt, 2001), among others. On a topic related to ours (Hardt, 1998) TBL has been used to add or remove right-peripheral material from the output of a VPE antecedent selection algorithm, for samples where the head of the antecedent was correctly identified but not the form.

There are a number of TBL implementations freely available, and I chose the μ-TBL system[21] (Lager, 1999). For comparison, a small number of experiments were replicated using fnTBL[22] (Ngai and Florian, 2001), and they gave very similar results.

2.2.2 Maximum entropy modelling

Maximum entropy models (MaxEnt) (Jaynes, 1957a; Jaynes, 1957b; Jaynes, 1965; Jaynes, 1967) (also called log-linear models, Gibbs models) use features, which can be arbitrarily complex, to provide a statistical model of the observed data which has the highest possible entropy. The model has maximal entropy in that it assigns the greatest likelihood to the events of the training corpus, but expresses no preference for any other conditions or events. Therefore, it models the events of this corpus and nothing else. This has an advantage over other statistical models such as Naive Bayes, where independence assumptions between features

[21]Downloadable from http://www.ling.gu.se/∼lager/mutbl.html - version 1.0.1
[22]Downloadable from http://nlp.cs.jhu.edu/ rflorian/fntbl/ - version 1.0

cannot usually be justified in NLP tasks. On the other hand, 'tweaking' of values to hardcode linguistic or domain-specific insights is not possible using MaxEnt. MaxEnt models differ from the transformation based learning algorithm described in subsection 4.2.2 in that a probability is returned as opposed to a binary outcome, and they do not produce easily readable rules as TBL does.

We will follow the intuitive description given by Berger et al. (1996, pp. 40-42). Taking as our problem the translation of the word *in* to French, a large set of decisions made by an expert is collected, finding that the translator always chooses one of the five French phrases : *dans, en, à, au cours de, pendant*. This information gives us the first constraint on the model $p(f)$, the probability that f is the translation of *in* :

$$p(dans) + p(en) + p(\grave{a}) + p(au\ cours\ de) + p(pendant) = 1$$

There are an infinite number of models satisfying this constraint, for example one where $p(dans) = 1$, or $p(pendant) = 1/2$ and $p(a) = 1/2$, but without further information these are not justifiable. Given only the information we have, the most defensible model intuitively is :

$$\begin{aligned} p(dans) &= 1/5 \\ p(en) &= 1/5 \\ p(\grave{a}) &= 1/5 \\ p(au\ cours\ de) &= 1/5 \\ p(pendant) &= 1/5 \end{aligned}$$

Studying the data available, it is further noticed that the translator chooses either *dans* or *en* in 30% of the cases. Now there are two constraints :

$$\begin{aligned} p(dans) + p(en) + p(\grave{a}) + p(au\ cours\ de) + p(pendant) &= 1 \\ p(dans) + p(en) &= 3/10 \end{aligned}$$

This gives the distribution below, in keeping with the philosophy of generating the most uniform model, and making no assumptions other than those given in the data :

$$\begin{aligned} p(dans) &= 3/20 \\ p(en) &= 3/20 \\ p(\grave{a}) &= 7/30 \\ p(au\ cours\ de) &= 7/30 \\ p(pendant) &= 7/30 \end{aligned}$$

Any number of constraints can be added to the model this way, and the features used can be as complex as desired.

Ratnaparkhi (1998a) makes a strong argument for the use of maximum entropy models, and demonstrates their use in a variety of NLP tasks, including POS tagging (1996), word-sense disambiguation, prepositional phrase attachment (1998b), sentence boundary detection (1997), and parsing (1997; 1999). To name some other applications, Rosenfeld (1996) shows improvements over trigrams in language modelling, Johnson et al. use MaxEnt for stochastic attribute-value grammars (1999), and Higgins and Sadock apply it to modelling scope preferences (2003).

Two different maximum entropy classifiers were used. The first one, the OpenNLP Maximum Entropy package[23] uses the Generalized Iterative Scaling (GIS) algorithm, and provides a simple form of smoothing (see section 2.2.7), in which features not seen in the training data are 'observed'.

The second package[24] is derived from the first, and uses the Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm (L-BFGS)[25]. This package also incorporates Gaussian Prior smoothing, which gives superior results for maximum

[23]Downloadable from https://sourceforge.net/projects/maxent/ - version 2.2.0

[24]Downloadable from http://www.nlplab.cn/zhangle/maxent_toolkit.html - version 20040315

[25]This package also has a GIS implementation, but it is derived from an earlier version of the OpenNLP package, and initial tests show that for the same settings it consistently performs slightly (1-2%) poorer.

entropy smoothing (Chen and Rosenfeld, 1999).

GIS and L-BFGS are two parameter estimation methods for maximum entropy. A comparison of these and others can be found in (Malouf, 2002), where it is argued that L-BFGS is superior. These two packages will be referred to as GIS-MaxEnt and L-BFGS-Maxent.

2.2.3 Decision Trees

A decision tree contains internal nodes which perform tests on the data, and following the result of these tests through to the end leaf gives the classification associated with the leaf. These methods have the advantage that they automatically discard any features that are not necessary, can grow more complicated tests from the tests (features) given, and that the resulting trees are usually humanly readable. Implementations of decision trees include CART (Breiman et al., 1984), ID3 (Quinlan, 1990), ASSISTANT (Cestnik et al., 1987) and C4.5 (Quinlan, 1993) (which is the one I used for my experiments[26]).

Decision trees have been used extensively in NLP, in tasks such as POS tagging (Cardie, 1993; Schmid, 1994; Orphanos et al., 1999), parsing (Magerman, 1994; Magerman, 1995; Haruno et al., 1998), feature selection for HPSG grammars (Toutanova and Manning, 2002), coreference resolution (McCarthy and Lehnert, 1995; Corston-Oliver, 2000; Soon et al., 2001), text categorization (Lewis and Ringuette, 1994), text summarization (Mani and Bloedorn, 1998), anaphora resolution (Aone and Bennet, 1996) and ellipsis resolution (Yamamoto and Sumita, 1999) among others.

The tree is constructed by finding the feature with the highest information gain ratio[27] and splitting the data with a test based on it. This procedure is repeated recursively, until tests cannot produce gains above a set threshold, or there are

[26]Downloadable from http://www.cse.unsw.edu.au/ quinlan/ - release 8

[27]Information gain measures how much a particular feature/question can reduce entropy, i.e. contribute to predicting the data. For example, if playing 20 questions and trying to guess a number between 1 and 100, the best question (barring, of course, a lucky guess) is whether the number is above or below 50. This question has the highest information gain, and produces the best split of the data. One problem with the information gain criterion is that it favours splits with many values, which can produce overly complex trees. Gain ratio tries to counteract this by penalizing multi-valued features.

too few values to split. Pruning of the tree can be performed afterwards, where questions that produce splits that may be statistically insignificant are collapsed, producing a single outcome. Following the construction of the tree, it is also possible to convert it into a series of *"If... Then..."* style, Horn-clause-like rules.

An important point to note is that decision trees are built using an inductive bias to generate the shortest tree that correctly classifies the training examples, following Ockham's razor. The argument is that a short hypothesis that fits the data is less likely to be a coincidence than a long one.

As an example of using decision trees C4.5 is trained on the data seen in Table 2.1 to generate a model to predict when it is a good day to play golf. The decision tree generated is seen in Figure 2.4, and the rules extracted from it using the C4.5rules program in Figure 2.5. It is seen that the temperature values are not used in the tree constructed, which may be because they are not informative or they overlap with a more informative feature.

Outlook	Temperature	Humidity	Windy	Play
sunny	85	85	false	Don't Play
sunny	80	90	true	Don't Play
overcast	83	78	false	Play
rain	70	96	false	Play
rain	68	80	false	Play
rain	65	70	true	Don't Play
overcast	64	65	true	Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
rain	75	80	false	Play
sunny	75	70	true	Play
overcast	72	90	true	Play
overcast	81	75	false	Play
rain	71	80	true	Don't Play

Table 2.1: Golf dataset for decision tree training

2.2.4 Memory Based learning

Memory based learning (Stanfill and Waltz, 1986; Daelemans, 1999) (MBL) (also known as instance-based learning, non-parametric regression) is a descendant of the classical k-Nearest Neighbour approach to classification. It takes the approach

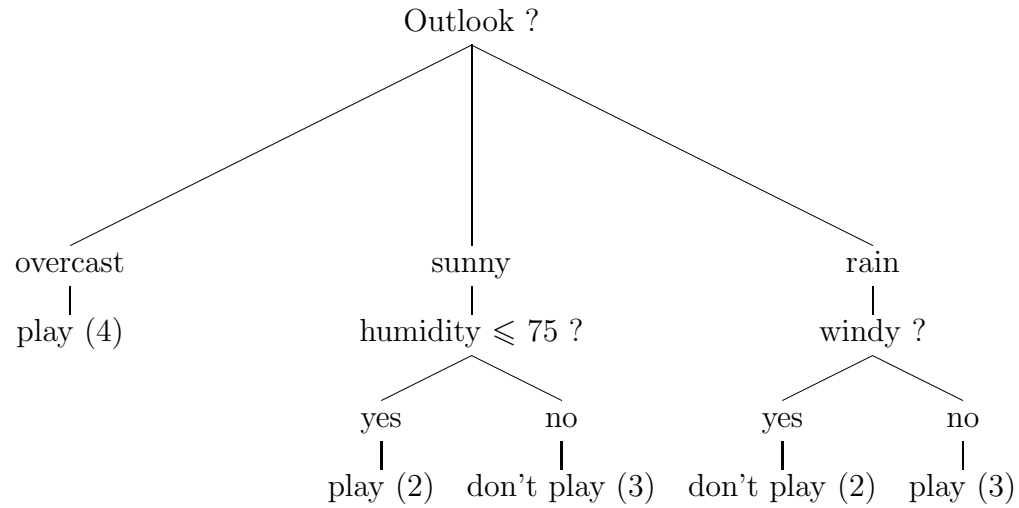

Figure 2.4: Decision tree for golf example

```
Rule 2:                          Rule 4:
        outlook = overcast               outlook = rain
        ->  class Play                   windy = false
                                         ->  class Play

Rule 1:                          Rule 3:
        outlook = sunny                  outlook = rain
        humidity > 75                    windy = true
        ->  class Don't Play             ->  class Don't Play

Default class: Play
```

Figure 2.5: Decision rules for golf example

of directly re-using previous experience rather than forming generalized rules from it. Learning is achieved through the storage of correct instances, and classification is achieved by extrapolation from similar instances in memory. Features are weighted according to a measure of significance (information gain, gain ratio etc.), and a combined score is calculated from neighbouring values, weighted by a distance metric.

The main feature separating MBL from other ML approaches is that all instances are stored and used, including noise and low-frequency events, and no attempt is made at generalizing or creating rules. It has been argued that this non-generalizing approach is representative of how certain processes function, such as

a doctor who diagnoses a patient by remembering a similar case and proceeding from there (Fix and Hodges, 1952). It has also been suggested (Daelemans et al., 1999b) that this type of learning is suitable for NLP tasks, as it is able to deal with exceptions and subregularities well, and its feature weighting produces smoothing effects, improving performance for sparse data.

MBL has been applied to a variety of NLP tasks, such as POS tagging (Daelemans et al., 1996; Zavrel and Daelemans, 1999), chunking (Veenstra, 1999; Veenstra, 1998), shallow parsing (Daelemans et al., 1999a), morphological analysis (den Bosch and Daelemans, 1999), word sense disambiguation (Hoste et al., 2002), parsing (Kubler, 2004), and classifying ellipsis types in dialogue (Fernández et al., 2004).

TiMBL (Daelemans et al., 2002)[28] was used for the experiments. TiMBL refines the MBL process defined above by ordering the training data in a loss-less decision-tree. This facilitates quicker classification, which is important as this is where the real processing in MBL happens - MBL is a 'lazy' learner, meaning a learner where learning involves only storing the training data.

2.2.5 SLIPPER

SLIPPER[29] (Simple Learner with Iterative Pruning to Produce Error Reduction) (Cohen and Singer, 1999) is a learning platform that uses boosting. Boosting (Meir and Ratsch, 2003) is a generalized method of improving the accuracy of any learning algorithm, which relies on the idea that a simple or "weak" learning algorithm, performing slightly better than chance, can be "boosted" to a "strong" algorithm. This is achieved by running the algorithm over the training set repeatedly, and each time increasing the weight on the examples that were not correctly identified before, forcing the weak learner to focus on these difficult cases. SLIPPER outputs a weighted set of rules, which are consulted for classification and give a combined final score.

SLIPPER operates by running a greedy learning algorithm on a subset of the training data, and pruning the rules on another subset. The learning algorithm

[28] Available from http://ilk.kub.nl/software.html#timbl - version 5
[29] Available from http://www-2.cs.cmu.edu/ wcohen/slipper/ - version 1 release 2.6

used is based on the popular AdaBoost (Freund and Schapire, 1999) algorithm. Experiments over a number of datasets show SLIPPER giving similar, slightly better performance than C4.5, and RIPPER, its predecessor (Cohen, 1995). SLIPPER boasts ease of interpretation with its Horn-clause-like rules, efficiency and scalability to large datasets among its advantages.

Applied to the problem set described in subsection 2.2.3 for decision trees, SLIPPER produces no rules, except defaulting to the class Play for all instances, which suggests that it requires larger datasets.

SLIPPER and RIPPER have been used for a variety of NLP applications, such as text classification (Cohen, 1996; Scott and Matwin, 1998), word segmentation (Meknavin et al., 1997), and recently, classifying ellipsis types in dialogue (Fernández et al., 2004).

2.2.6 Ensemble learning

Ensemble learners combine a number of classifiers to improve accuracy. An example of this is SLIPPER, a boosting algorithm (described in Section 2.2.5) . SLIPPER belongs to the family of ensemble learners that use several versions of the same classifier in combination to increase performance. Another popular method in this family is Bagging (Breiman, 1996a), where classifiers trained on different samples of the training set are combined in a voting scheme. Arcing (Breiman, 1996b) combines ideas from these two methods, where the data set is sampled, as in bagging, but the likelihood of a data point being in the sampled sets is weighted according to mistakes by earlier classifiers, reminiscent of boosting.

The other group of ensemble learners focus on combining different types of base classifiers. This is of interest to us, as we will be comparing results from multiple classifiers for experiments. The most obvious way of achieving this is a simple voting scheme, where usually a majority vote is needed for the ensemble to accept a prediction.

Another method employed for combining classifiers is stacked generalization, or *stacking* (Wolpert, 1992). In stacking, a number of base learning methods are used as input data to another learning algorithm. The data to be used is di-

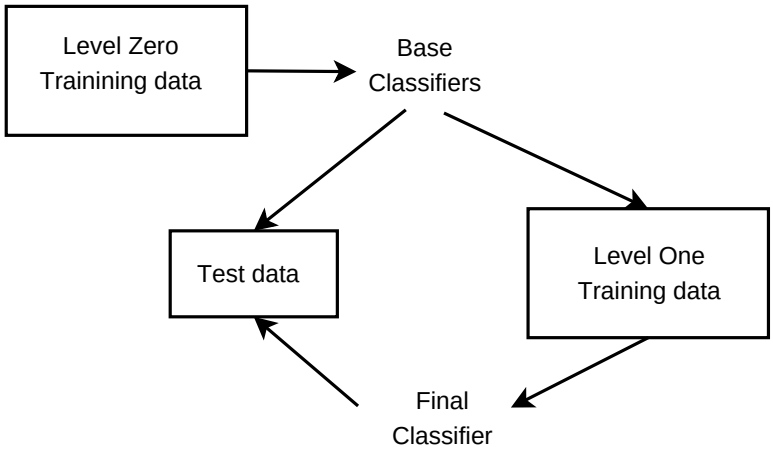

Figure 2.6: Stacking

vided into 3; the 'level zero' training data, 'level one' training data and the final test data. The base classifiers are trained on the level zero data, and predict the level one and test data, as illustrated in Figure 2.6. Their predictions are incorporated as features into the sets they predict. The enriched level one data is then used by another classifier as training data, and the final results are achieved by testing this model on the test data. Stacking can be performed with binary predictions (yes/no) or with confidence estimates (Ting and Witten, 1999), if the base classifiers support them, with the latter method usually yielding better results (Dzeroski and Zenko, 2004).

I will experiment with both voting and stacking, with the expectation that the different learners will complement each other.

2.2.7 Effects of data size

A crucial factor in the performance of machine learners is, of course, the training data. Data size is important because natural language shows properties of a Zipfian distribution (Zipf, 1949). Zipf's law states that for a variety of domains, the frequency of an event divided by its rank is constant, i.e. the second most frequent event would be one half the frequency of the most frequent, the third one third and so on. This holds for distributions ranging from city populations to word frequencies in a corpus.

Examining the word frequencies for an unfinished draft of this thesis, for example, reveals that the most common word ('the') was seen 1451 times, and the second most common ('of') 886 times. These frequencies drop quickly however, and 45% of the words encountered were only seen once, and 62% less than twice. These results are summarised in Figure 2.7, with the figure on the left plotted on normal axes, and the figure on the right plotted on logarithmic axes.

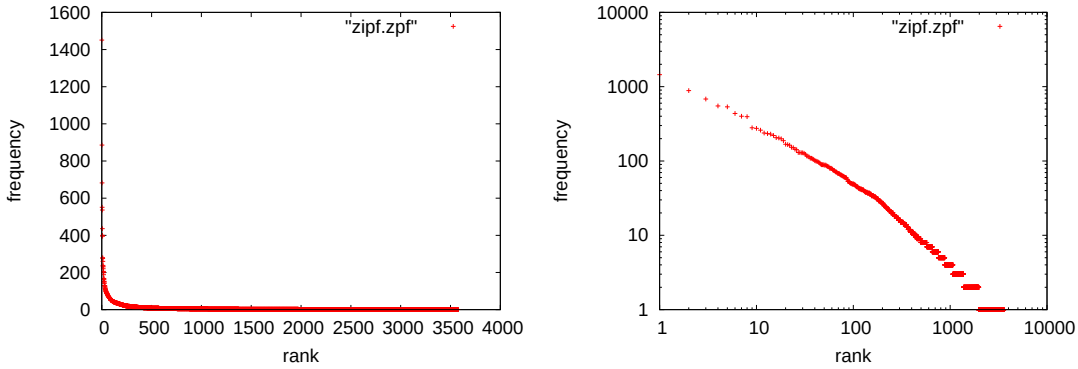

Figure 2.7: Zipf curves

It follows from this distribution of events that a small number of events are observed very frequently, but a large number of events are observed very rarely, if at all, for any given corpus. This presents a major obstacle to machine learning approaches, in the form of unseen events. Using naive methods will result in models where unseen events are given zero probability, resulting in low performance. There are three ways of dealing with this problem, which can be combined.

The first is to aim for a bigger corpus, as doubling data size should halve the number of unseen events. Unfortunately, even with very large corpora, there will still be unseen events, although fewer. It follows that performance is strongly correlated with data size. Furthermore, the validity of results obtained using small corpora may be questionable, as Banko and Brill (2001) show, given that performance gains with very large increases in training data size are considerable. For the task of confusion set disambiguation, where the correct use of a word must be chosen from a set of words with which it is commonly confused (ie {to,two,too}, {then, than}), they show, using a number of machine learning algorithms, that scores of around 82% accuracy with a 1 million word training corpus increase to more than 95% accuracy with 1000 million words.

Unfortunately, compiling training sets of this size is not currently possible for many NLP tasks, including mine. My training corpus will remain far short of the standard that Banko and Brill recommend, given that VPE occurs with very low frequency, and a tradeoff had to be made between the time spent looking for VPEs and experiments. I do, however, feel that the size of the corpora, at over 1.2 million words containing over 1500 VPE samples when combined, is large enough to give credible initial answers to the questions posed in this thesis.

The second method is to adopt a more generalizing approach as the principle of a machine learner. Rule-based systems such as TBL, decision trees and SLIPPER provide such an advantage. The rules seen in Figure 2.5 can successfully deal with instances not found in the original training set, as the rules learned capture generalizations and not explicit mappings.

The third method is smoothing. Smoothing (Good, 1953; Chen and Goodman, 1996) allocates some of the probability from seen events, which are likely to be overestimated by naive ML approaches, to unseen events. Considering the classifiers employed, the extent to which smoothing is used varies. Maximum entropy models do not employ smoothing by themselves, but both the packages that I use have their implementations of smoothing; TBL, decision trees and SLIPPER don't use smoothing, but because they extract rules, they are robust to noise. MBL uses feature weighting, which has a limited smoothing effect.

2.2.8 Fine tuning the algorithms

It is important to note that while a number of machine learning approaches were used, the aim of this work is not to compare these approaches, but to investigate the tenability of corpus-based, statistical VPE resolution. Except for setting thresholds for the MaxEnt methods, optimizing the smoothing for one, and weighting/decimating SLIPPER and C4.5, I have tried to use each ML package as-is, with a minimum amount of experimentation with its considerable combination of settings. It is possible that some increases in performance could be obtained through the optimization of such settings, but as the aim is to produce a general system, this is not necessary, and fine-tuning can be done if the system needs to be optimized to a particular domain.

Banko and Brill (2001) find that with large enough corpora the difference in performance between competing machine learning algorithms tends to diminish, and the gains possible through increases in training data dwarf those achieved through experimentation with different settings.

Chapter 3

Experimental method and data

3.1 Assessing performance

3.1.1 Assessing VPE detection

Each classification falls under one of four possibilities :

	VPE	Not VPE
Classified VPE	True Positive (TP)	False Positive (FP)
Classified not VPE	False Negative (FN)	True Negative (TN)

The performance for the experiments is calculated using recall, precision, and the F1-measure, defined below :

$$\text{Recall} = \frac{\#\text{True positives}}{\#\text{True positives} + \#\text{False negatives}} \tag{3.1}$$

$$\text{Precision} = \frac{\#\text{True positives}}{\#\text{True positives} + \#\text{False positives}} \tag{3.2}$$

The F1 provides a measure that combines these two at a 1/1 ratio.

$$\text{F1} = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3.3}$$

Recall measures how many of the cases of VPE being looked for are found by the classifier. Precision measures what ratio of those identified as elliptical by the classifier actually are. The F1 measure gives the harmonic mean of these two. The F-measure can be adapted to reward recall or precision more, but I will take them to be equally important for this task.

Given a test set of 100 VPEs, if an algorithm identifies 80 of them correctly (True Positives), but returns a further 40 which are not actually VPE (False Positives), recall=80/100=80%, precision=80/(80+40)=66.67%, and F1=72.72%.

The accuracy measure, which uses success averaged over both positive and negative cases, is not used, as the negative cases have too large a majority. For the VPE detection experiments described in section 4.3, for example, using the Treebank data, there are over 140 thousand negative cases (auxiliaries that are not VPE), and only around 600 positive cases (auxiliaries that are VPE). Given this ratio, simply ignoring VPEs and forming a "baseline" that labels all instances as not VPE would achieve 99.55% accuracy, with an error rate of 0.45%. With only 0.45% improvement possible overall, it would be difficult to judge the results of experiments.

Another consequence of the ratio of positive to negative examples is that scores for negative examples will not be reported, as recall, precision and F1 would always be close to 100% and non-informative[1].

In cases where data is being enriched across consecutive experiments, two figures derived from the F1 will also be presented to ease interpretation :

$$\text{Absolute Error Reduction } (|\text{ER}|) = \text{F1}_{new} - \text{F1}_{old} \tag{3.4}$$

$$\text{Percentage Reduction in Error } (\%\text{ER}) = \frac{|ER|}{100 - \text{F1}_{old}} \tag{3.5}$$

[1]If an experiment yields 400 True Positives and 100 False Positives for the positive samples using the dataset mentioned above, the scores would be 66.67% Recall, 80% Precision and 72.73% F1. The scores for the negative samples, with 139900 True Positives and 200 False Positives, would be 99.93% Recall, 99.86% Precision and 99.89% F1. Another experiment, in which the positive samples get 500 True Positives and 100 False Positives, would have 83.33% Recall, Precision and F1 for the positive samples, which is a large change. The scores for the negative samples would all be 99.93%, however, which is too small a change to reflect the improvement between experiments.

|ER| is a useful way of keeping track of improvements, but %ER can give better insights. For instance, an experiment that improves performance from 60% F1 to 70%, and another one that improves it from 70% to 80% have the same |ER|. However, the first gives a %ER of 25%, while the second gives 33.3%. In many NLP tasks, once a certain level of performance is reached, efforts for improvement meet increasingly diminishing returns, and it can be argued that improvements after this point become more significant. The %ER measure gives a more accurate representation in these cases.

Initially, held-out data experiments were conducted where the data is split into training and test sections. This allows for quick results, and is useful to develop and optimize features. Final results, however, were obtained using cross-validation. N-fold cross validation is achieved by dividing all of the data randomly[2] into N sets, and then using one as test set while the rest combined are the training set, and repeating this for each of the N sets. The results for each experiment thus performed are weighted and averaged to give the final score.

Cross-validation has the advantage of minimising the effects of a particular split of training/test data, which may produce artificially high or low results. The final score obtained is more robust and reliable. I use 10-fold cross-validation for evaluation, which is fairly common in practise. The reason this method is not used for all the experiments is the cost in time it incurs, and also that the held-out method gives a limited portion of the data that can be analyzed and used to develop the features. Using the complete dataset would be problematic and lead to overestimation of performance, in effect negating the gains and methodological reasons for using cross-validation. It is true that having development data completely separate would be even more desirable, but given that this section is less than one third of the data (in terms of VPEs), their random distribution in the 10 sections of the cross-validation should ensure minimum adverse effects.

[2] Using stratified cross-validation instead, which aims to divide the data such that an equal numbers of instances for each class is observed across folds, could also have been considered here, but it was decided that the data was large enough to result in representative folds using random sampling. The stratification could also be extended to generate folds containing representative numbers of each type of VPE described in Chapter 6, but this was also decided against as the rare cases would be spread too thin to learn from, while the common cases could be distributed equally well using random sampling.

In section 4.3, where a number of versions of the data are run, with an increasing number of features, statistical significance testing is employed. It has been suggested (Dietterich, 1998) that McNemar's χ^2 test (Everitt, 1977) is appropriate to test large datasets where binary results are dependent. This test has a low likelihood of reporting significance in the differences when there is none. For successive experiments where we want to test if the added features significantly improve results, the pattern of agreements and disagreements between the classifiers is extracted :

	Classifier 1 mistake	Classifier 1 correct
Classifier 2 mistake	A	B
Classifier 2 correct	C	D

The intuition behind this test is to only look at the samples in the test where the decisions of the classifiers are different (counts B and C), and ignore the samples where their decisions agree (counts A and D). Statistical significance only exists if there is a sufficient quantity of instances belonging to the first category. It can be inferred that the aim here is to disregard the performance of the classifiers and instead focus on their differences only. It is, for example, possible for two classifiers having near identical performance in terms of F1 to be significantly different, as they classify different samples differently, while generating the same number of True and False Positives.

The null hypothesis is that neither of the classifiers significantly outperforms the other. For this, C must be equal to B. McNemar's test employs a χ^2 distribution based on the formula :

$$\frac{(|B-C|-1)^2}{B+C} \tag{3.6}$$

If the computed result is below $\chi^2_{1,0.95} = 3.844159$, the χ value for a significance of 0.05, the null hypothesis holds. If the result is above this, the distribution is significant. If it is above 6.64, it has a significance of 0.01, and if it is above 10.83, of 0.001.

It should be noted that while McNemar's test is robust against detecting a difference when there is none, it does not measure variability arising from the choice

of the training set. The choice of the training data may influence whether or not significant difference is found between two classifiers. Cross-validation tests can be more reliable in these instances.

3.1.2 Assessing antecedent selection

Performance will be measured using Hardt's criteria of Head Overlap, Head Match, and Exact Match with human choice, providing three levels of success. These are defined as :

- Head Overlap : The system is successful if either the head verb of the system choice is contained in the human annotation choice, or the head verb of the human annotation choice is contained in the system choice.
- Head Match : The system is successful if the system choice and human annotation choice have the same head verb.
- Exact Match : The system is successful if the system choice and human annotation choice match word for word.

The following examples from Hardt illustrate the differences between these criteria :

(32) When bank financing for the buy-out collapsed last week, so did UAL's stock.

- Coder : collapsed
- Algorithm : collapsed last week

Here, the result is unsuccessful according to Exact Match, but successful according to Head Match and Head Overlap.

(33) By contrast, in 19th-century Russia, an authoritarian government owned the bank and had the power to revoke payment whenever it chose, much as it would in today's Soviet Union.

Two choices for the antecedent appear :

- Coder : revoke payment whenever it chose
- Algorithm : owned the bank and had the power to revoke payment whenever it chose

Here, the result would be unsuccessful according to Exact Match and Head match, but successful according to Head Overlap.

The statistical significance test employed for the VPE detection experiments are not used for the antecedent location experiments as the benchmark performance is high, and the improvements made in consecutive experiments on it are not large enough to be significant.

3.1.3 Assessing resolution

For the classification of VPEs into the type of resolution needed, the Recall, Precision and F1 measures will be used for each class. An aggregate score will also be produced using the counts for all events.

For the resolution step, a simple string comparison will be made between human annotated resolved sentences and those generated by the program. The result will be given as the percentage of successful sentences.

Statistical significance tests are not applicable here as there are no consecutive experiments.

3.2 Data used

3.2.1 Annotation methodology

Annotation was performed by reading through the selected texts completely, twice, and with a further check searching for all auxiliaries and reading the sentences they occur in. The annotation was stored in a stand-off format, in a

separate file from the source data, with one line per VPE instance. An example is seen below.

```
wsj_00.520.57       ->520.36-45 | Tense-base | Composer Marc
Marder , a college friend of Mr Lane 's who earns his living
playing the double bass in classical music ensembles , has
prepared an exciting , eclectic score that tells you what the
characters are thinking and feeling far more precisely than
intertitles , or even words , would tell you what the
characters are thinking and feeling .
```

This example contains 4 pieces of information:

1. wsj_00.520.57 : This means the VPE site was found in the WSJ corpus, section 00. It is in line 520[3], and in that line, word 57 from the beginning.

2. 520.36-45 : The antecedent for the VPE was also found in sentence 520, and starts at word 36 and ends at word 45, inclusive.

3. Tense-base : This is the category assigned for the resolution of this VPE (as described in Chapter 6); for successful resolution, the verb in the antecedent would have to be changed to the base tense.

4. Composer Marc ... : The final part is what the resolved sentence should look like.

The annotation was performed by a single annotator, as the process is time-consuming and resources were not available for a second annotator. It should be noted, however, that perfect agreement was found with Daniel Hardt for the cases considered for this work on the intersection corpus described in 3.2.3, .

All sections chosen for annotation were selected by random choice.

[3]All subsections of sections are concatanated to produce a single file per section for the Treebank data

3.2.2 Data used in VPE detection

Taking as our aim high accuracy combined with simplicity, a series of experiments were done with increasing amounts of information, resulting in three sets of corpora. The British National Corpus (BNC) (Leech, 1992)[4] was used for the first round of experiments. It contains 100 million words, of which 90% are written text, and the rest spoken dialogue. It is annotated with Part of Speech (POS) tags, using the CLAWS-4 tagger (Leech et al., 1994)[5].

Initial experiments were done using the held-out data method, where a set portion is allocated for training, and the rest for testing. A range of sections of the BNC, containing around 370k words[6] with 645 samples of VPE were used as training data. The separate test data consists of around 74k words[7] with 197 samples of VPE.

The sections chosen from the BNC are all written text and consist of extracts from novels, autobiographies, scientific journals and plays. The average frequency of VPE occurrences for the whole data is about once every 525 words, or once every 32 sentences.

To experiment with what gains are possible through the use of more complex data in the form of parse trees, the Penn Treebank (Marcus et al., 1994a; Marcus et al., 1994b)[8] was used for the second round of experiments. It contains 1 million words, consisting of 1989 Wall Street Journal material and parts of the Brown Corpus, annotated in Treebank II style.

The Penn Treebank has more than a hundred phrase labels, and a number of empty categories, but it uses a coarser tagset[9] than the BNC. A mixture of sections from the Wall Street Journal and Brown corpus were used. The training section[10] consists of around 540k words and contains 517 samples of VPE. The test section[11] consists of around 140k words and contains 150 samples of VPE.

[4]More information available at http://www.natcorp.ox.ac.uk
[5]Details in Appendix A
[6]Sections CS6, A2U, J25, FU6, H7F, HA3, A19, A0P, G1A, EWC, FNS, C8T
[7]Sections EDJ, FR3
[8]More information available at http://www.cis.upenn.edu/treebank/home.html
[9]Details in Appendix B
[10]Sections WSJ 00, 01, 03, 04, 15, Brown CF, CG, CL, CM, CN, CP
[11]Sections WSJ 02, 10, Brown CK, CR

The WSJ sections of the Treebank have a very low rate of VPE occurrence at one VPE every 1750 words, or once every 77 sentences. This is understandable, given that using ellipsis is 'bad form' for journalists. The Brown sections, on the other hand, have a VPE every 617 words, or 38 sentences, more like the BNC. Overall, the marked up data has a VPE every 1000 words, and every 45 sentences.

The next set of experiments uses the corpora mentioned above, but strip POS and parse information, and parses them automatically. This is both to overcome training data limitations, and more importantly, to make the system more robust and independent of corpus availability. These experiments and the parsers used are discussed in section 4.4.

For VPE detection experiments, all verbs are treated as potential VPEs. The reason for including all verbs and not only those identified as auxiliaries is to ensure that all positive examples are always available, even if incorrect tagging occurs using automatic parsers (or manually annotated data, albeit rarely).

3.2.3 Data used in antecedent location

The data used for the antecedent location experiments is the same as for the VPE detection experiments. Because the version of the Treebank used in previous work (see section 5.1) is different from the one used in the current experiments, to be able to directly compare the results, the intersection between the corpus used in the previous work and our test corpus was used.

The result is an intersection corpus of 58 samples, out of the 150 in the whole test corpus. Experiments are done on this corpus, on both the old version of the Treebank using the original VPE-RES implementation (see Section 5.2), and on the current version of the Treebank using the new implementation. This allows us to verify that no mistakes have been made in the re-implementation.

Experiments are then made on the whole test set, but the test set is mainly used to analyze errors. The final results are computed using all of the training and test data combined for the baseline approach, and using cross-validation for the machine learning approaches. Experiments are repeated for the parsed data.

3.2.4 Data used in resolution

The data used for the antecedent resolution experiments is the same as for the previous two sets of experiments. The work presented in Chapter 6 consists of two parts : in the first part, statistics on the types of VPE found data and the level of difficulty of their resolution is presented; in the second part experiments with an automatic classifier and resolution module are described.

For the work on the first part of the chapter, all the data from both the Treebank and BNC datasets was used, and classification decisions were made. A rule-based classifier is built and tested on the Treebank data. Experiments are repeated with parsed data from both datasets. Unlike the previous two experiments, however, only one of the parsers presented in Section 4.4.1 is used for the resolution experiments. This is because phrase-level information is not used in the automatic classification process, and a comparison using both will reveal little of interest. Because trace information is required, only the Charniak parser was used for parsed data experiments.

Chapter 4

Detection of VPEs

This chapter describes work done on the first stage of the VPE algorithm. Previous work is described, then experiments are conducted with increasing amounts of information, allowing the effects of each part of the information to be assessed as it is added.

Section 4.2 describes experiments done with the BNC, using only word form and POS data. A manual baseline is constructed, and experiments are done with five machine learning algorithms to improve results : TBL, GIS-MaxEnt, L-BFGS-MaxEnt, C4.5, MBL (see section 2.2 for detailed discussion).

Section 4.3 describes experiments done with the Penn Treebank, which contains parse-structure information as well as word form and POS data. Experiments are done with three machine learning algorithms used in Section 4.2 (MBL and the two versions of Maximum entropy), and another one is introduced, SLIPPER.

Section 4.4 describes experiments done using both BNC and Treebank data, but stripped of the POS tags, and automatically parsed. Experiments are done with the four machine learning algorithms used in Section 4.3.

4.1 Previous work

While the theoretical study of VPE detection is quite extensive, the only empirical experiment done for this task to date, to our knowledge, is with Hardt's (1997)

algorithm for detecting VPE in the Penn Treebank. Unlike the present work, Hardt includes cases such as comparatives and 'do so/it/that' anaphora as VPEs.

This work looks for the pattern, (VP (-NONE- *?*)), in the Treebank to detect VPEs, relying on the parse annotation having identified empty expressions correctly. Evaluated against a manual search for VPEs done on 3.2% of the data (155k words), it achieves recall levels of 53% and precision of 44%, giving an F1 of 48% using a simple search technique which relies on the annotation having identified empty expressions correctly.

It should be noted that while Hardt's results are low, his main focus was on antecedent location, and VPE detection was not a step he gave a lot of weight to. Therefore, his results should be seen as preliminary.

Low performance in this first stage could lead to systematic errors being introduced, such as VPEs in a certain context being ignored repeatedly, or non-elliptical verbs being accepted as elliptical, given a certain context. Such systematic errors can produce incorrect conclusions through analysis; so it becomes clear that an initial stage with higher performance is necessary.

Earlier versions of the work described in this section can be found in Nielsen (2003a; 2003b; 2004d; 2004a; 2004c; 2004b). Minor differences with these earlier results are present, due to experiments at different stages using different versions of the classifiers, updated as they are released. Repeated cross-validation experiments are also bound to produce slight differences, due to the random distribution of the data sets. None of these differences are large, and they do not affect any of the conclusions drawn.

4.2 Experiments using the BNC

4.2.1 Baseline approach

It is desirable to develop a VPE-detection algorithm that can perform well using only POS and lexical information, as currently POS tagging is possible with much higher accuracy than full, or even partial parsing. This means that methods developed for a corpus such as the BNC will not experience large losses in

performance when applied to automatically tagged data.

A simple heuristic approach was developed to form a baseline. 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. The method also does a short backwards search for verbs. The forward search looks 7 words ahead and the backwards search 3. Both stop at sentence boundaries, and skip 'asides', which are taken to be snippets between commas without verbs in them, such as : "... papers do, however, show ...".

The algorithm was optimized on the development data, and achieves recall of 89.60% and precision of 42.14%, giving an F1 of 57.32%.

4.2.2 Transformation-based learning

As the precision of the baseline method is not acceptable, the use of machine learning techniques is investigated, beginning with transformation based learning (see section 2.2.1), a flexible and powerful learning algorithm.

Generating the training samples is straightforward for this task. We trained the μ-TBL system using the words and POS tags from BNC as the 'initial guess'. For the gold standard we replaced the tags of verbs which are elliptical with a new tag, 'VPE'. Two example sentences from the training data are seen in Figure 4.1[1].

4.2.2.1 Generating rule templates

The learning algorithm needs to have the rule templates specified in which it can search. As an initial experiment, we used a sample set of rule templates that was included in the μ-TBL distribution, the templates used by Brill to train a POS tagger, which use the 3 word neighbourhood context, but in a limited way (see footnote on page 46).

[1]Descriptions of the BNC tags can be found in Appendix A.

Word	POS	gold-POS
I	PNP	PNP
mean	VVB	VVB
I	PNP	PNP
would	VM0	**VPE**
,	PUN	PUN
but	CJC	CJC
you	PNP	PNP
would	VM0	**VPE**
nt	XX0	XX0
.	PUN	PUN

Figure 4.1: Input data to TBL

The results of training the system with these templates is seen in Table 4.1, where the threshold is the value new rules need to satisfy in number of net improvements; when this fails, the algorithm stops learning. Lower thresholds mean more rules are learned, but also increase the likelihood of spurious rules being acquired. With a threshold of 5, 22 rules are learned. Lowering the threshold to 3 results in a total of 38 rules being learned.

Threshold	Recall	Precision	F1
5	48.13	70.07	57.06
3	51.87	70.70	59.84

Table 4.1: Results with simple POS tagging rule templates

The top 10 rules learned are seen in Table 4.2. The second column shows the score of the rule, which is how many corrections it made to resemble the gold standard more. The third column is which tags it changes, and the fourth column what tag it changes them into. The last column indicates the conditions for the rule to be applied; for the first rule in the table, this means that the rule is applied only if the current word is tagged VDD (did) and if one of the next 2 words is tagged as 'PUN' (puncuation mark); the second rule is applied if the current word is tagged VM0 (modal auxiliary), the previous word is tagged as 'PNP' (personal pronoun) and the next word 'PUN'.

It can be seen that the first and third rules learned by the TBL algorithm with the given templates are rather simple; if a word with tag 'VDD' or 'VDB' occurs one

Rank	Score	Change	to	if
1	50	VDD	VPE	tag[1,2]=PUN
2	29	VM0	VPE	tag[-1]=PNP and tag[1]=PUN
3	28	VDB	VPE	tag[1,2]=PUN
4	28	VBZ	VPE	tag[-1]=PUN and tag[1]=XX0
5	26	VM0	VPE	tag[1]=PNP and tag[2]=PUN
6	20	VM0	VPE	tag[1]=XX0 and tag[2]=PUN
7	11	VDB	VPE	wd[0]=do and wd[2]=you
8	11	VPE	VDB	tag[1,2,3]=VVI
9	11	VDD	VPE	tag[-1]=CJS
10	10	VHB	VPE	tag[1]=PNP and tag[2]=PUN

Figure 4.2: Top 10 rules learned by Brill's POS tagging templates

or two words before a punctuation mark, it's a VPE. This reflects the corpus in that a majority of the instances of VPE are indeed found at the end of sentences. Many of the other rules encode sequences such as 'He can' (rule 2,), 'did he ?' (rule 5).

The only rule that makes it to the top 10 which corrects spurious VPE tags introduced by previous rules is rule 8: if any of the three following words is an infinitive lexical verb, it changes the VPE tag back to 'VDB', if it was incorrectly transformed by rule 3.

Adding some more extended templates, up to 10 words ahead and behind, experiments were repeated. It must be noted that this extension is simplistic and consists of a search for a single tag or word in the 5 to 10 word neighbourhood as an indication for VPE, but does not include any permutations. Using these templates, 45 rules are learned for a threshold of 5, and 60 for a threshold of 3.

Threshold	Recall	Precision	F1
5	57.01	81.88	67.22
3	59.81	80.00	68.45

Table 4.2: Results with simple POS tagging templates extended with handwritten templates

To make the learning process more independent of biases in the patterns, it is desirable to have a larger set of templates which are not handcrafted. Defining five

features, {tag[-1,-2,-3]$_a$, wd[-1,-2,-3]$_b$, wd[0]$_c$, tag[1,2,3]$_d$, wd[1,2,3]$_e$}, templates were generated based on each of their permutations[2] . The results of using these templates is seen in Table 4.3, with 64 rules learned for a threshold of 5, and 71 for a threshold of 3.

Threshold	Recall	Precision	F1
5	31.31	72.83	43.79
3	37.85	69.23	48.94

Table 4.3: Results with grouped neighbourhood templates

As the grouped templates do not give such good results, we tried generating templates based on all permutations of {tag[-2], tag[-1], tag[+1], tag[+2], wd[-2], wd[-1], wd[0], wd[+1]}. We would have liked to do this over a larger context, but the number of permutations gets too large for the learning algorithm, which runs out of memory. The results with these templates are seen in Table 4.4. 49 rules are learned for a threshold of 5, and 62 for a threshold of 3.

Threshold	Recall	Precision	F1
5	55.14	74.21	63.27
3	57.94	72.09	64.25

Table 4.4: Results with permuted neighbourhood templates

Rank	Score	Change	to	if
1	63	VDD	VPE	tag[1,2,3]=PUN and wd[0]=did
2	58	VBZ	VPE	tag[1]=XX0 and tag[2]=PNP and wd[0]=is
3	50	VM0	VPE	tag[1]=PUN and wd[1]=.
4	42	VDB	VPE	tag[1,2,3]=PUQ and wd[0]=do
5	69	VPE	VDD	tag[1,2,3]=VVI and wd[0]=did
6	38	VM0	VPE	tag[1]=PNP and tag[2]=PUN and wd[1]=you
7	34	VM0	VPE	tag[1]=XX0 and wd[1]=nt
8	24	VDD	VPE	tag[-1]=CJS and wd[0]=did
9	20	VBB	VPE	tag[1]=PNP and tag[2]=PUN and wd[-1]=,
10	20	VBB	VPE	tag[1]=XX0 and tag[2]=PNP and wd[-1]=,

Figure 4.3: Top 10 rules learned by combined templates

[2]The templates range from a, a & b, a & c and so on up to a & b & c & d & e.

Combining all the templates discussed so far, the rules seen in Figure 4.3 and the results in Table 4.5 are obtained. As the recall is the lower figure, we tried to increase it. It can be seen in Tables 4.2, 4.3 and 4.4 that lowering the threshold increases recall, but reduces precision. Modifying the rules learned by removing those which correct possibly spuriously tagged VPEs[3] cause recall to be increased, but again at a cost to precision, as seen in the rows with the 'modified' attribute set to 'yes' in Table 4.5. The decrease in precision is larger than the increase in recall, lowering the overall F1 score.

Threshold	Modified	Recall	Precision	F1
5	no	50.93	79.56	62.11
5	yes	53.73	70.55	61.00
3	no	**62.15**	**75.56**	**68.20**
3	yes	65.42	67.96	66.67

Table 4.5: Results for initial TBL

4.2.2.2 POS grouping

Despite the fact that the training data consists of 370k words, there are only around 650 elided verbs in it. The sparseness of the data limits the performance of the learner, so a form of smoothing which can be incorporated into the transformation based learning model is needed. To achieve this, auxiliaries were grouped into subcategories of 'VBX', 'VDX' and 'VHX', where 'VBX' generalizes over 'VBB', 'VBD' etc. to cover all forms of the verb 'be'; 'VHX' generalizes over the verb 'have' and 'VDX' over the verb 'do'. Rules learned after this grouping was specified are seen in Figure 4.4. Using the combined templates discussed in the previous section, 54 rules are learned for a threshold of 5, and 93 for a threshold of 3.

The results of this grouping on performance are seen in Table 4.6. Both precision and recall are increased, with F1 improving by 5% or 12% - the effect is larger for the higher threshold[4].

[3]Rules which change a 'VPE' tag to something else, such as the rule seen in the fifth rule in Figure 4.3

[4]It may be noted that modifying the rules learned does not change the recall for this experiment, but this is a coincidence; while the numbers are the same, there are differences in the samples of ellipses found.

Rank	Score	Change	to	if
1	92	VBX	VPE	tag[1]=XX0 and tag[2]=PNP and wd[-1]=,
2	60	VDX	VPE	tag[1]=AV0 and wd[1]=so
3	50	VM0	VPE	tag[1]=PUN and wd[1]=.
4	50	VBX	VPE	tag[1]=PNP and tag[2]=PUN and wd[-1]=,
5	50	VDX	VPE	tag[1]=PUN and wd[0]=did
6	38	VM0	VPE	tag[1]=PNP and tag[2]=PUN and wd[1]=you
7	36	VDX	VPE	tag[1]=XX0 and tag[2]=PUN and wd[1]=nt
8	36	VDX	VPE	tag[1]=XX0 and tag[2]=PNP and wd[-1]=,
9	34	VDX	VPE	tag[1]=PNP and tag[2]=PUN and wd[-1]=,
10	32	VM0	VPE	tag[1]=XX0 and tag[2]=PUN and wd[1]=nt

Figure 4.4: Top 10 rules learned by partially grouped TBL

Threshold	Modified	Recall	Precision	F1	|ER|	%ER
5	no	**68.22**	**82.02**	**74.49**	12.38	32.67
5	yes	68.22	79.78	73.55	12.55	32.17
3	no	68.69	79.03	73.50	5.30	16.66
3	yes	68.69	76.56	72.41	5.74	17.22

Table 4.6: Results for partially grouped TBL

To further alleviate the data sparseness, all auxiliaries were then grouped to a single POS tag 'VPX'. Using these templates, 45 rules are learned for a threshold of 5, and 60 for a threshold of 3. Rules learned after this full grouping is given are seen in Figure 4.5. Some rules which were nearly identical in Figure 4.4 are combined now, such as rules 7 and 10 becoming rule 6 in Figure 4.5. The rules which are learned by the system are still simple, as some examples of when they work illustrated below show :

- '[He laughed], did/didn't he ?' (rules 1/2)
- 'As did [his wife]' (rule 9)

The performance of the system increases even more with the extended grouping, although not as much as with the initial grouping step, as seen in Table 4.7. These experiments suggest that for the task at hand, the initial POS tag distinctions are too fine grained, and the system benefits from the smoothing achieved through grouping.

Rank	Score	Change	to	if
1	150	VPX	VPE	tag[1]=XX0 and tag[2]=PNP and wd[-1]=,
2	130	VPX	VPE	tag[1]=PNP and tag[2]=PUN and wd[-1]=,
3	86	VPX	VPE	tag[1]=XX0 and tag[2]=PUN and wd[1]=nt
4	64	VPX	VPE	tag[-1]=PNP and wd[1]=.
5	36	VPX	VPE	tag[1]=AV0 and tag[2]=PUN and wd[1]=so
6	34	VPX	VPE	tag[1]=XX0 and tag[2]=PUN and wd[1]=n't
7	32	VPX	VPE	tag[1]=PUM and wd[0]=did
8	30	VPE	VPX	tag[-1,-2,-3]=DTQ
9	26	VPX	VPE	tag[-1]=CJS and wd[0]=did
10	26	VPX	VPE	tag[-2]=PNP and tag[-1]=AV0 and wd[1]=.

Figure 4.5: Top 10 rules learned by fully grouped TBL

Threshold	Modified	Recall	Precision	F1	|ER|	%ER
5	no	**69.63**	**85.14**	**76.61**	2.12	8.31
5	yes	71.96	78.57	75.12	1.57	5.94
3	no	71.03	82.61	76.38	2.88	10.87
3	yes	73.36	76.96	75.12	2.71	9.82

Table 4.7: Results for fully grouped TBL

For the best F1, the system can achieve recall of 69.6% and precision of 85.1%. Tilting the balance in favour of recall increases it to 73.4%, but reduces precision to 77%. Here, as in most of the experiments, modifying the rules results in a decrease of F-score by about 1-1.5%.

4.2.2.3 Problems with TBL

While experiments using TBL have produced promising results, these are not optimal due to limitations within the μ-TBL package itself. Trying to overcome this problem by using another TBL package, fnTBL, failed, as it also couldn't handle the data. As with most machine learning tasks, the performance of the system increases with larger training data. It has not been possible to increase the training data size to more than 370k words with the templates used. Conversely, it has not been possible to increase the permutation and range of templates without decreasing the size of the training corpus. This is not necessarily due to flaws in the training algorithm as such, but more a result of the fact that the

implementation was simply not designed for this kind of use and cannot handle the memory requirements. The training data size could have been increased for the other algorithms, but was not for comparison purposes.

4.2.3 Maximum entropy modelling

These experiments are done using the two systems described in subsection 2.2.2.

4.2.3.1 Feature selection

Maximum entropy allows for a wide range of features, but for initial experiments only word form and POS information will be used. (34a) shows a sentence with VPE, and (34b) shows the information passed to the learning algorithm for the three word neighourhood of the elliptical verb. (34c) shows a sentence without VPE, and (34d) shows the arguments passed for the non-elliptical verb. *w(n)* is the verb being checked, *t(n)* its POS tag, and at the end of the line TRUE/FALSE signifies whether it is elliptical or not.

(34) a. The party divisions between leading reformers such as Lloyd George, Mosley and Macmillan also hindered { co-operation, as **did** personal hostility within } political parties.

b. w(n-3)=co-operation t(n-3)=NN1 w(n-2)=, t(n-2)=PUN w(n-1)=as t(n-1)=CJS w(n)=did t(n)=VDD w(n+1)=personal t(n+1)=AJ0 w(n+2)=hostility t(n+2)=NN1 w(n+3)=within t(n+3)=PRP TRUE

c. { The measurements **were** made with the } system described in this article.

d. w(n-2)=The t(n-2)=AT0 w(n-1)=measurements t(n-1)=NN2 w(n)=were t(n)=VBD w(n+1)=made t(n+1)=VVN w(n+2)=with t(n+2)=PRP w(n+3)=the t(n+3)=AT0 FALSE

Experiments with different amounts of forward/backward context give the results seen in Table 4.8. Context here is defined as the number of words to each side of

the auxiliary being considered, limited by sentence boundaries. For each sample being checked, the classifier returns a probability for the outcome being elliptical. The threshold for accepting the results for a potential VPE were set to 0.2, or 20%; this value is just an initial guess formed by looking at the first couple of results.

	GIS-MaxEnt			L-BFGS-MaxEnt		
Context size	Recall	Precision	F1	Recall	Precision	F1
1	67.75	42.02	51.87	65.42	42.94	51.85
2	76.16	53.79	63.05	78.03	58.39	66.79
3	**72.43**	**61.26**	**66.38**	**74.76**	**66.66**	**70.48**
4	64.48	60.00	62.16	64.01	64.92	64.47
5	63.08	63.38	63.23	37.38	56.73	45.07
6	59.81	64.64	62.13	32.71	54.68	40.93
7	57.47	62.43	59.85	31.77	62.38	42.10
8	53.73	59.89	56.65	31.30	57.75	40.60
9	51.86	61.32	56.20	33.17	57.25	42.01
10	50.00	60.45	54.73	33.64	64.86	44.30
15	48.13	59.53	53.22	42.99	64.78	51.68
20	45.79	62.82	52.97	27.10	61.70	37.66

Table 4.8: Effects of context size on maximum entropy learning

It is seen that L-BFGS-MaxEnt gives the highest performance, with an F1 of 70.48% compared to GIS-MaxEnt's 66.38%, for a context size of 3. The results show that for large contexts the algorithms show reduced performance. This is due to the fact that the contexts do not allow for the kind of generalization available to transformation based learning. After a certain point, the effect of the context size levels, as few sentences are larger than 10 or 15 words to each side of an auxiliary verb. It is also observed that GIS-MaxEnt degrades more gracefully, without huge drops in performance as seen for L-BFGS-MaxEnt when context size is increased from 4 to 5. This may be due to the fact that using the default settings GIS-MaxEnt performs 100 iterations of modeling, while L-BFGS-MaxEnt performs only 30, and a higher number of iterations may be needed for a large number of features.

To determine the relative importance of forward vs backward context, experiments were performed discarding each (but keeping the current word in both cases) with the results seen in Tables 4.9 and 4.10. It is seen that forward con-

text is far more important than backward context, and that for the backward context the immediately previous word is the most important. Again, L-BFGS outperforms GIS by a large margin. For the case of a 3-word context L-BFGS performs as well without the backwards context as with it, giving high recall and low precision without the backwards context, and a more balanced recall and precision using both backward and forward context.

	GIS-MaxEnt			L-BFGS-MaxEnt		
Context size	Recall	Precision	F1	Recall	Precision	F1
1	41.12	30.87	35.27	40.65	30.85	35.08
2	81.77	51.16	62.94	81.77	54.01	65.05
3	**77.10**	**54.45**	**63.82**	**81.77**	**62.50**	**70.85**
4	73.36	56.27	63.69	80.84	58.24	67.71
5	71.96	58.33	64.43	78.50	61.31	68.85
6	70.09	57.03	62.89	78.03	62.31	69.29
7	66.82	57.20	61.63	75.23	60.98	67.36
8	65.88	56.85	61.03	77.57	60.14	67.75
9	63.55	55.73	59.38	77.10	60.21	67.62
10	63.08	56.01	59.34	72.89	62.90	67.53

Table 4.9: Effects of forward context on maximum entropy learning

	GIS-MaxEnt			L-BFGS-MaxEnt		
Context size	Recall	Precision	F1	Recall	Precision	F1
1	**30.37**	**33.33**	**31.78**	**33.17**	**36.41**	**34.71**
2	22.42	24.12	23.24	30.37	32.66	31.47
3	21.42	21.19	21.34	34.57	32.03	33.25
4	23.36	22.22	22.77	29.90	26.55	28.13
5	21.49	21.59	21.54	30.37	26.20	28.13
6	20.56	23.40	21.89	28.03	24.89	26.37
7	18.69	21.97	20.20	26.16	25.33	25.74
8	18.69	23.95	20.99	23.83	24.05	23.94
9	17.75	23.45	20.21	16.35	26.92	20.34
10	16.82	22.92	19.40	25.23	23.89	24.54

Table 4.10: Effects of backward context on maximum entropy learning

4.2.3.2 Thresholding

Setting the forward/backward context size to 3, experiments are run to determine the correct setting for the threshold. This value is used to determine at what

level of confidence from the model a verb should be considered a VPE.

	GIS-MaxEnt			L-BFGS-MaxEnt		
Threshold	Recall	Precision	F1	Recall	Precision	F1
0.1	78.50	47.86	59.46	78.97	60.79	68.69
0.15	76.16	54.88	63.79	76.63	65.07	70.38
0.2	72.43	61.26	66.38	74.76	66.66	70.48
0.25	**68.22**	**65.17**	**66.66**	73.36	67.96	70.56
0.3	63.08	67.16	65.06	72.42	70.13	71.26
0.35	59.34	70.16	64.30	**71.96**	**72.98**	**72.47**
0.4	57.00	71.76	63.54	71.02	73.78	72.38
0.45	52.80	73.85	61.58	70.56	74.38	72.42
0.5	49.53	74.64	59.55	67.28	74.61	70.76

Table 4.11: Effects of thresholding on maximum entropy learning

With higher thresholds, recall decreases as expected, while precision increases (Table 4.11). For GIS, F1 peaks at 0.25, which is close to the initial guess of 0.2. The fact that this value is so low is expected to be due to the size of the corpus. For subsequent experiments, a threshold of 0.2 will be retained, for comparison purposes, and because its results are very close to those of 0.25.

For L-BFGS, a higher threshold of 0.35 gives the highest performance. For subsequent experiments, this will be the threshold used for this algorithm, as it offers a 2% improvement in F1 over a 0.2 threshold.

4.2.3.3 POS Grouping

Using the same principles for smoothing introduced in section 4.2.2.2, the effects of category grouping are investigated. This is accomplished by changing the data input to the format seen in (35), with VDX or VPX replacing the original POS tag, depending on the level of grouping. For GIS-MaxEnt, Table 4.12 shows an increase in F1 of 2.5% for a context size of 3, using partial grouping. Full grouping, seen in Table 4.14 gives a further 2% increase.

(35) Replacement grouping :

w(n-3)=and t(n-3)=CC w(n-2)=when t(n-2)=WRB w(n-1)=he t(n-1)=PRP w(n)=did **t(n)=VDX/VPX** w(n+1)=comma t(n+1)=comma w(n+2)=he t(n+2)=PRP w(n+3)=vowed t(n+3)=VBD TRUE

For L-BFGS, partial grouping gives a 1.41% F1 increase for a context size of 3 (Table 4.13), but full grouping reduces this by 0.98% (Table 4.15). This suggests that for this algorithm, the information added is only marginally more useful than the information removed.

It is interesting to note that the |ER| effect of grouping for GIS-MaxEnt is less than that for transformation based learning; 4% compared to 8%. Furthermore, seen from the perspective of %ER, grouping only gives a 12% %ER for GIS-MaxEnt and practically none for L-BFGS, while transformation based learning gets a 38% %ER.

Context size	Recall	Precision	F1	|ER|	%ER
2	76.63	53.77	63.19	0.14	0.38
3	**73.83**	**64.48**	**68.84**	2.46	7.32
4	68.69	61.76	65.04	2.88	7.61
5	64.48	63.59	64.03	0.8	2.18

Table 4.12: Results for partially grouped GIS-MaxEnt

Context size	Recall	Precision	F1	|ER|	%ER
2	76.16	65.72	70.56	-	-
3	**73.36**	**74.40**	**73.88**	1.41	5.12
4	1.96	74.03	72.98	-	-
5	71.02	75.62	73.25	-	-

Table 4.13: Results for partially grouped L-BFGS-MaxEnt

Context size	Recall	Precision	F1	|ER|	%ER
2	77.57	55.14	64.46	1.27	3.45
3	**76.16**	**65.72**	**70.56**	1.72	5.52
4	67.28	64.00	65.60	0.56	1.60
5	64.95	67.47	66.19	2.16	6.01

Table 4.14: Results for fully grouped GIS-MaxEnt

Using full grouping brings the performance of GIS closer to that of L-BFGS; 70.5% compared to 72.9%. While maximum entropy gives better results than baseline, it scores lower than TBL.

Context size	Recall	Precision	F1	|ER|	%ER
2	83.64	60.06	69.92	-0.64	-2.17
3	**71.02**	**74.87**	**72.90**	-0.98	-3.75
4	57.94	65.26	61.38	-11.6	-42.93
5	58.41	71.02	64.10	-9.15	-34.21

Table 4.15: Results for fully grouped L-BFGS-MaxEnt

4.2.3.4 Smoothing

GIS-MaxEnt provides a simple method of smoothing, in which features that weren't seen in the training data are 'observed'. L-BFGS-MaxEnt, on the other hand, provides Gaussian prior smoothing, which has been shown to give superior results among maximum entropy smoothing methods (Chen and Rosenfeld, 1999).

Smoothing	GIS-MaxEnt			L-BFGS-MaxEnt		
parameter	Recall	Precision	F1	Recall	Precision	F1
0.1	72.89	09.35	16.58	18.69	100.00	31.49
0.2	69.62	10.23	17.84	52.80	90.40	66.66
0.3	68.69	10.56	18.30	64.01	84.04	72.67
0.4	67.28	11.05	18.98	70.09	81.52	75.37
0.5	64.95	11.26	19.19	72.42	79.48	75.79
0.6	62.61	11.30	19.15	73.36	77.72	75.48
0.7	61.68	11.55	19.46	**75.23**	**77.40**	**76.30**
0.8	**59.34**	**11.74**	**19.61**	75.70	76.05	75.87
0.9	57.47	11.66	19.40	75.70	76.05	75.87

Table 4.16: Effects of smoothing on maximum entropy learning

Although the results for the two sets of experiments are presented side by side in Table 4.16, the smoothing parameters applied to them are unrelated, but share the same range. The smoothing for GIS-MaxEnt does not produce useful results. This is probably due to the fact that this smoothing technique is still in an experimental stage, and may become more useful in the future. L-BFGS with Gaussian prior smoothing, on the other hand, improves F1 by 3.4%, a 12.5% %ER. This brings it to the same level of performance as TBL.

4.2.4 Decision tree learning

The C4.5 algorithm, described in subsection 2.2.3. works in two steps: first, a large tree is constructed that creates simple rules for every example found, and second, more generalized rules are extracted from this tree. Running both parts of the algorithm, C4.5 deduces that the best rule to learn is that there is no need to recognize VPEs, and everything is non-elliptical, as this results in only a 1.4% error overall. This fits with C4.5's design, which is to not overfit data, and produce few, general rules[5].

The data available to C4.5 was exactly the same as the data used for the Maximum entropy models. Regardless of choice of grouping level or context size, the resulting trees ignored VPEs. To counteract the weighting given to non-elided verbs, we experimented with removing non-elided samples from the training corpus. Even with this, using both stages of the algorithms results in overgeneralization, but results after only the first part of the algorithm are now useable.

Table 4.17 shows the effects of decimation on partially grouped data with a context size of 3. Decimating at a rate n operates by discarding every nth non-elided verb. The columns show different decimation settings. The reason decimation of only up to 10 is shown is because at higher rates the effect of the decimation vanishes and the classifier resorts to ignoring VPE again. Experiments also show that higher context sizes than 3 do not produce any different results, meaning that the extra data is too noisy for the classifier to model.

Decimation	3	5	8	10
F1	42.23	28.89	28.89	-

Table 4.17: Effects of decimation for partially grouped data using decision tree learning, context size 3

With full grouping, seen in Table 4.18, better results are obtained, but the use of context sizes above 5 does not provide any improvements. Decimation rates between 20 and 40 all give the same results, and at higher values than this result in overgeneralization. Looking at the best result obtained, at context size 5 and

[5] It has been pointed out that using the Gain ratio criterion instead of Information gain while constructing the tree can mitigate this problem, as this has been shown to give improved performance on small datasets. Investigation of this will be left for future work.

decimation of 20, it is seen that the algorithm obtains precision of 79.39% and recall of 60.93%, giving an F1 of 68.94%.

C/D	3	5	8	10	15	20
3	57.55	67.62	67.62	67.89	67.87	68.25
5	58.82	68.25	68.25	68.58	68.55	**68.94**

Table 4.18: Effects of context size and decimation for fully grouped data using decision tree learning

It can be observed that while performance decreases on non-grouped data as decimation is increased, the reverse holds for grouped data. While this may at first seem surprising, it can be explained by the fact that results for the non-grouped experiments are rather volatile, and difficult to draw conclusions from.

4.2.5 Memory Based learning

TiMBL, described in section 2.2.4. was trained with the same data used for the maximum entropy and C4.5 experiments. It is seen that a context size of 3 again gives the best results (Table 4.19), and that MBL achieves good results with the level of information used.

Context size	Recall	Precision	F1
1	51.40	53.39	52.38
2	71.49	69.23	70.34
3	**73.83**	**72.14**	**72.97**
4	72.42	69.50	70.93
5	71.49	70.18	70.83
6	70.56	72.24	71.39
7	70.56	66.51	68.48

Table 4.19: Results for MBL

4.2.5.1 POS grouping

Using the same principles for smoothing introduced in section 4.2.2.2, the effects of category grouping are investigated. Table 4.20 shows a 0.6% decrease in F1 for a context size of 3, using partial grouping. Full grouping (Table 4.21), on the

other hand, gives a 2.5% increase. MBL is seen to benefit little from grouping, but the overall effect is still positive and will be retained.

Context size	Recall	Precision	F1	|ER|	%ER
1	50.00	51.69	50.83	-1.55	-3.25
2	71.49	65.94	68.60	-1.74	-5.87
3	**74.76**	**70.17**	**72.39**	-0.58	-2.15
4	72.42	68.88	70.61	-0.32	-1.10
5	72.42	69.81	71.10	0.27	0.93
6	71.49	69.54	70.50	-0.89	-3.11
7	71.02	65.23	68.00	-0.48	-1.52

Table 4.20: Results for partially grouped MBL

Context size	Recall	Precision	F1	|ER|	%ER
1	50.93	54.50	52.65	1.82	3.70
2	74.29	68.24	71.14	2.54	8.09
3	**76.16**	**73.75**	**74.94**	2.55	9.24
4	73.83	70.53	72.14	1.53	5.21
5	73.36	69.77	71.52	0.42	1.45
6	72.89	70.90	71.88	1.38	4.68
7	73.36	67.09	70.08	2.08	6.50

Table 4.21: Results for fully grouped MBL

4.2.6 Cross-validation

Ten-fold cross-validation is performed on the algorithms. It is not possible to perform it for TBL and C4.5 due to memory problems. MBL and GIS-MaxEnt have 3% lower F1 than they did on the test data, and L-BFGS-MaxEnt has 0.3% lower F1. These results are quite consistent with the experiments on the development set and help verify the conclusions drawn.

Algorithm	Recall	Precision	F1
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 4.22: Cross-validation on the BNC

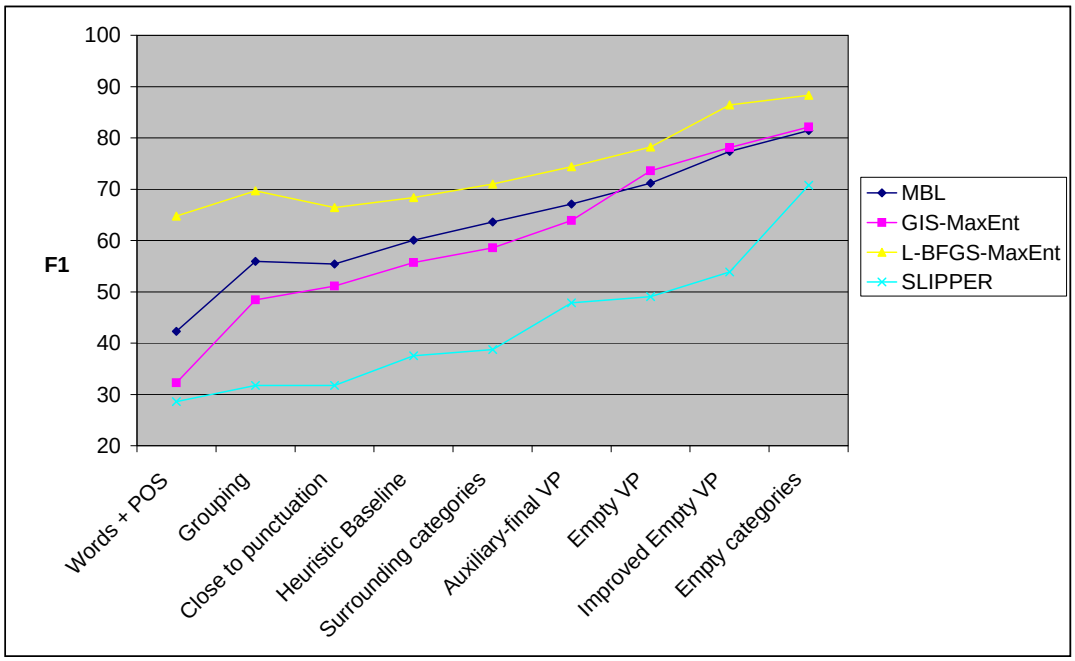

Figure 4.6: F1 plot for algorithms on Treebank data versus features being added

4.3 Experiments using the Penn Treebank

To determine 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.

Experiments in the previous section showed that TBL and decision trees are not suitable for the task at hand, for different reasons. μ-TBL, due to internal limitations, cannot handle large data/templates, and C4.5 is not designed for sparse data, and has problems with large data sizes as well. This leaves us with MBL and maximum entropy, but another rule based learner, SLIPPER, is added.

Figures 4.6, 4.7 and 4.8 can be consulted for comparative viewing of F1 scores, error reduction and percentage error reduction, as the experiments progress. Using only POS tags and word forms, results vary between 33% and 62% F1. Adding features based on syntactic information, final results ranging from 64% and 82% are achieved. SLIPPER consistently performs the worst, and L-BFGS-MaxEnt is consistently the best.

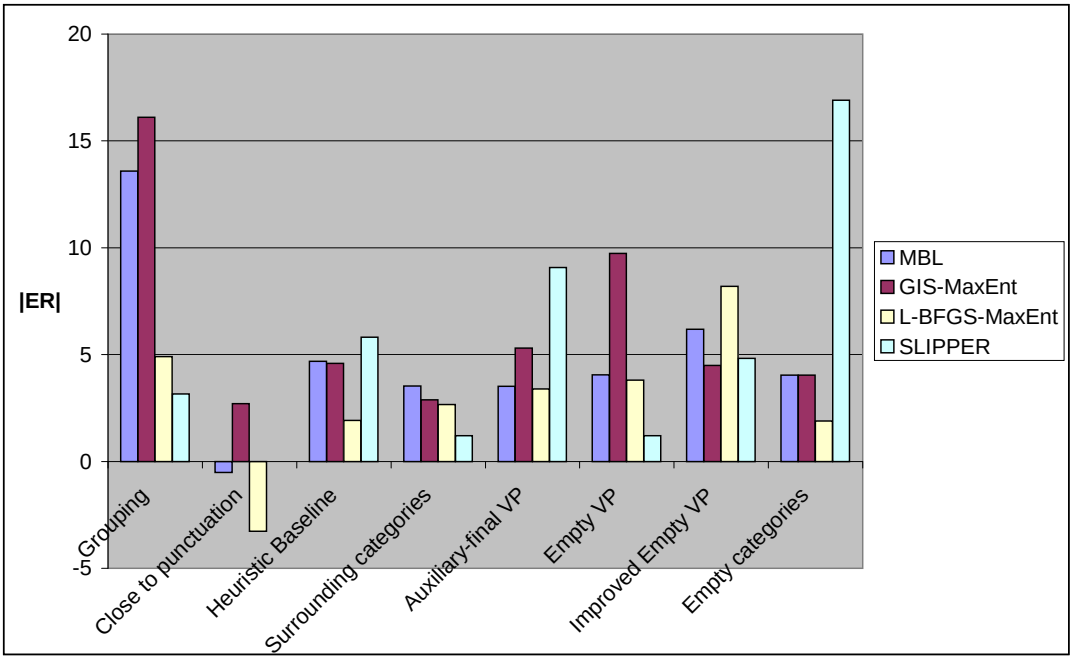

Figure 4.7: Error Reduction effect of features on Treebank data

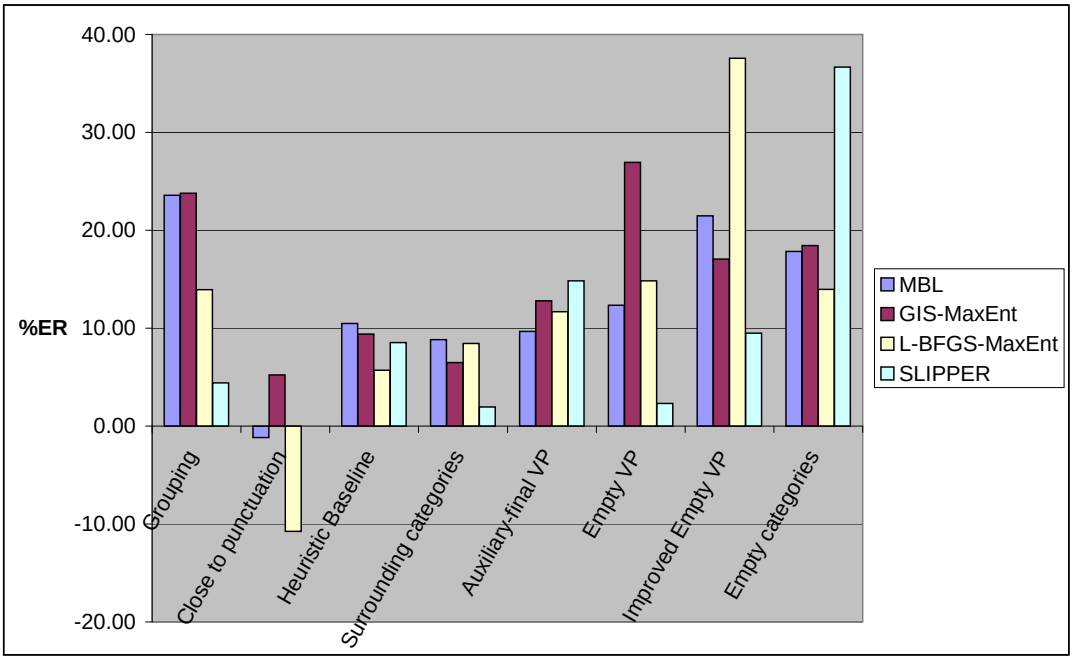

Figure 4.8: Percentage Error Reduction effect of features on Treebank data

4.3.1 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 4.23 are seen to be considerably poorer than those for the BNC, despite the comparable data sizes. While stylistic differences in the corpora may account for some of this, it is more likely due to the coarser tagset employed. The algorithms decline at different rates, with the worst being GIS-MaxEnt. It is again seen that the smoothing employed in the L-BFGS implementation has quite a large effect.

Algorithm	Recall	Precision	F1
MBL	40.52	44.28	42.32
GIS-MaxEnt	20.26	79.48	32.29
L-BFGS-MaxEnt	52.28	85.10	64.77

Table 4.23: Initial results with the Treebank

4.3.2 SLIPPER

At this stage, experiments were done with another algorithm, SLIPPER, described in subsection 2.2.5. SLIPPER brings a greedy rule-based learner into the experiments once again, as the two rule-based systems used in previous experiments were dropped.

Like the C4.5 algorithm, SLIPPER does not seem to be particularly suited to data with a skewed balance of positive/negative samples, and gives poorer results than the other algorithms (Table 4.24). It is possible to counteract these limitations to some degree by weighting the elliptical samples, using a feature built into SLIPPER. A sample which is given a weight of 5, for example, is treated as if it had occurred 5 times when deriving rules, and since positive samples are under-represented in the data set, experiments were done with giving these a higher weight than 1. Best results are achieved by weighting 6 to 15 times; as these all give the same results, a mid-point of 10 times weighting will be used for subsequent experiments. An objection to this method may be that this artificial weighting, which has a purpose similar to the decimation done for the decision

tree algorithm in section 4.2.4, results in a corpus that is not representative in the way an un-weighted corpus is. However, it must be kept in mind that this is simply a way of counter-acting a bias towards majority data inherent in the algorithm. Although SLIPPER does not perform well, it produces readable rules, and will be retained for further experiments.

Weighting	Recall	Precision	F1
1	2.92	60.00	7.36
2	7.19	50.00	12.57
3	13.07	51.28	20.83
4	17.65	51.92	26.34
5	19.61	44.12	27.15
6-15	**21.57**	**42.31**	**28.57**
20	41.18	13.32	20.13
30-50	49.02	8.96	15.15

Table 4.24: Results using the SLIPPER algorithm

For a weighting of 10, SLIPPER produces 7 rules to classify a verb as VPE, seen in Figure 4.9, with the default being false. The rules learned work together to generate an aggregate score, and they have different weights on the outcome. They are rather simple so far, and mostly discover that VPE happens to auxiliary and modal verbs.

Rank	VPE if
1	word[-2]=as and tag[-1]=PRP(personal pronoun)
2	word=did
3	word=does
4	word[+3]=?
5	word=do
6	tag[+1]=.
7	tag[+1]=MD(modal verb)

Figure 4.9: Rules learned by SLIPPER

Fernandez et al. (2005) perform experiments on classifying ellipsis types in dialogue, using both MBL and SLIPPER, and achieve good performance with both. This suggests that it is not a flaw in the SLIPPER algorithm that is responsible for the low results, but that it is not suitable for the type of data at hand.

4.3.3 POS Grouping

At this stage, experiments were done with grouping in two different ways : as replacement or as added data. The difference is illustrated in the following example:

(36) a. Original sample :

w(n-3)=and t(n-3)=CC w(n-2)=when t(n-2)=WRB w(n-1)=he t(n-1)=PRP w(n)=did t(n)=VBD w(n+1)=comma t(n+1)=comma w(n+2)=he t(n+2)=PRP w(n+3)=vowed t(n+3)=VBD TRUE

b. Replacement grouping :

w(n-3)=and t(n-3)=CC w(n-2)=when t(n-2)=WRB w(n-1)=he t(n-1)=PRP w(n)=did **t(n)=VPX** w(n+1)=comma t(n+1)=comma w(n+2)=he t(n+2)=PRP w(n+3)=vowed t(n+3)=VBD TRUE

c. Added data grouping :

w(n-3)=and t(n-3)=CC w(n-2)=when t(n-2)=WRB w(n-1)=he t(n-1)=PRP w(n)=did t(n)=VBD w(n+1)=comma t(n+1)=comma w(n+2)=he t(n+2)=PRP w(n+3)=vowed t(n+3)=VBD **VB=NonVBX VD=VDX VH=NonVHX VP=VPX** TRUE

So far, replacement grouping has been used in experiments. Added data grouping, as it is seen here, incorporates the two levels of information that were experimented with before, partial and full grouping, while retaining the information present in the original tag of the verb itself.

Performing the two methods of grouping gives the results seen in Tables 4.25 and 4.26. For MBL, both forms of grouping give considerable improvement, but replacement grouping more so than added data. For GIS and SLIPPER replacement gives little improvement, while added data gives better results - much better in the case of GIS-MaxEnt.

For LBFGS, replacement actually results in a decrease in performance, and added data in an improvement. This is in keeping with previous results, where LBFGS

did not benefit greatly from replacement grouping, perhaps due to the smoothing already in use nullifying improvements from the smoothing of the grouping. Interestingly, while replacement grouping gives a significant difference according to the McNemar test, added data grouping does not. The reason for the added data grouping not showing statistical significance, despite a bigger change in F1 than replacement grouping, can be seen by looking at its raw counts. The grouping gives 20 more correctly identified VPEs, but also 20 more spurious guesses. For the McNemar test this means that the error rate stays the same, while for precision, recall, and therefore F1, correctly identifying VPEs is more important than correctly identifying non-VPEs.

It should be noted that for SLIPPER using replacement grouping is highly significant, despite virtually identical performance according to the other measures. The reason for this is that the actual classifiers built are indeed different, and while SLIPPER arrives at the same score, it does so with quite different classifications.

Experiments from this point on for MBL will continue to use replacement grouping, but maximum entropy and SLIPPER experiments will use added data grouping. It should be also noted that on this corpus grouping is more useful for MBL than it was on the BNC, which may be due to the benefits of grouping now being higher than the loss incurred in the original categories, which are now less informative.

Algorithm	Recall	Precision	F1	|ER|	%ER	Significance
MBL	**50.98**	**61.90**	**55.91**	13.59	23.56	0.001
GIS-MaxEnt	24.18	63.79	35.07	2.78	4.11	none
L-BFGS-MaxEnt	52.28	72.72	60.83	-3.94	-11.18	0.01
SLIPPER	44.44	21.12	28.63	0.06	0.08	0.001

Table 4.25: Replacement grouping results

Algorithm	Recall	Precision	F1	|ER|	%ER	Significance
MBL	47.71	60.33	53.28	10.96	19.00	0.001
GIS-MaxEnt	**34.64**	**80.30**	**48.40**	16.11	23.79	0.01
L-BFGS-MaxEnt	**65.35**	**74.62**	**69.68**	4.91	13.94	none
SLIPPER	**28.10**	**36.44**	**31.73**	3.16	4.42	none

Table 4.26: Added data grouping results

4.3.4 Close to punctuation

A very simple feature that checks for auxiliaries close to punctuation marks was tested. Table 4.27 shows the performance of the feature itself, characterized by very low precision, and results obtained by using it. It gives a small increase in F1 for GIS-MaxEnt, has no effect on SLIPPER, but gives a small decrease for L-BFGS-MaxEnt and 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 size of the development set can contribute to fluctuations in results.

While this feature is not one that is unambiguously useful, and is statistically insignificant for all classifiers, it does have the capacity to aid performance, and will be retained.

Algorithm	Recall	Precision	F1	|ER|	%ER	Significance
close-to-punctuation	30.06	2.31	4.30			
MBL	50.32	61.60	55.39	-0.52	-1.18	none
GIS-MaxEnt	37.90	78.37	51.10	2.7	5.23	none
L-BFGS-MaxEnt	59.47	75.20	66.42	-3.26	-10.75	none
SLIPPER	28.10	36.44	31.73	0	0	none

Table 4.27: Effects of using the close-to-punctuation feature

4.3.5 Heuristic Baseline

The baseline developed for the BNC (see Subsection 4.2.1) is adapted to the Treebank, and used as a feature. Its performance is considerably lower on the Treebank dataset (Table 4.28), which can be explained by the coarser tagset employed by the Treebank, compared to the tagset of the BNC. While both MaxEnt classifiers show no significant change, this feature does provide 5-10% %ER.

Algorithm	Recall	Precision	F1	|ER|	%ER	Significance
heuristic	48.36	27.61	35.15			
MBL	55.55	65.38	60.07	4.68	10.49	0.01
GIS-MaxEnt	43.13	78.57	55.69	4.59	9.39	none
L-BFGS-MaxEnt	62.09	76.00	68.34	1.92	5.72	none
SLIPPER	66.01	26.23	37.55	5.82	8.52	0.001

Table 4.28: Effects of using the heuristic feature

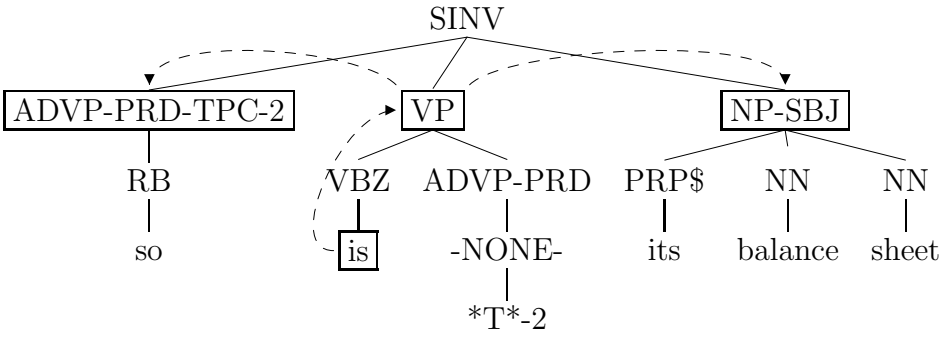

Figure 4.10: Fragment of sentence from Treebank illustrating the surrounding categories

4.3.6 Surrounding categories

The next features added are the head categories of the previous branch of the tree, and the next branch. So in the example in Figure 4.10[6], 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 4.29, giving a 1.2-3.5% boost, but this is largely statistically insignificant.

Algorithm	Recall	Precision	F1	|ER|	%ER	Significance
MBL	58.82	69.23	63.60	3.53	8.84	none
GIS-MaxEnt	45.75	81.39	58.57	2.88	6.50	0.05
L-BFGS-MaxEnt	64.05	79.67	71.01	2.67	8.43	none
SLIPPER	61.44	28.31	38.76	1.21	1.94	none

Table 4.29: Effects of using the surrounding categories

[6]Parse trees will be represented as graphs where possible, to facilitate interpretation, but will be copied verbatim from their corpus where they are too large to be displayed this way.

4.3.7 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), with about 31% precision[7], which is consistent with my findings (Table 4.30).

While this feature gives improvements to all classifiers, the changes are for the most part not large enough to be statistically significant.

Algorithm	Recall	Precision	F1	|ER|	%ER	Significance
Auxiliary-final VP	72.54	35.23	47.43			
MBL	63.39	71.32	67.12	3.52	9.67	none
GIS-MaxEnt	54.90	76.36	63.87	5.3	12.79	none
L-BFGS-MaxEnt	71.24	77.85	74.40	3.39	11.69	none
SLIPPER	75.82	34.94	47.84	9.08	14.83	0.05

Table 4.30: Effects of using the Auxiliary-final VP feature

4.3.8 Empty VP

Hardt (1997) uses a pattern check to search for empty VP's identified by the Treebank, '(VP (-NONE- *?*))', where '*?*' signals elided material. It achieves over 98% precision (Table 4.31), making it an excellent feature, as it has no drawbacks - every time it fires it's correct. 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, and, to a minor extent, the fact that Hardt includes comparatives in his analysis and we don't. This feature improves results considerably (Table 4.31).

It was observed that this search may be too restrictive to catch some examples of VPE in the corpus, such as the one in seen in Figure 4.11. It also misses examples of pseudogapping, as seen in Figure 4.12.

Modifying the search pattern to be '(VP (-NONE- *?*)', which is a VP that

[7]This result is for a search that looks for auxiliary final VPs and auxiliary final sentences, as previous versions of the Treebank would sometimes fail to insert a VP. This is fixed now, and the current search is done only for VPs.

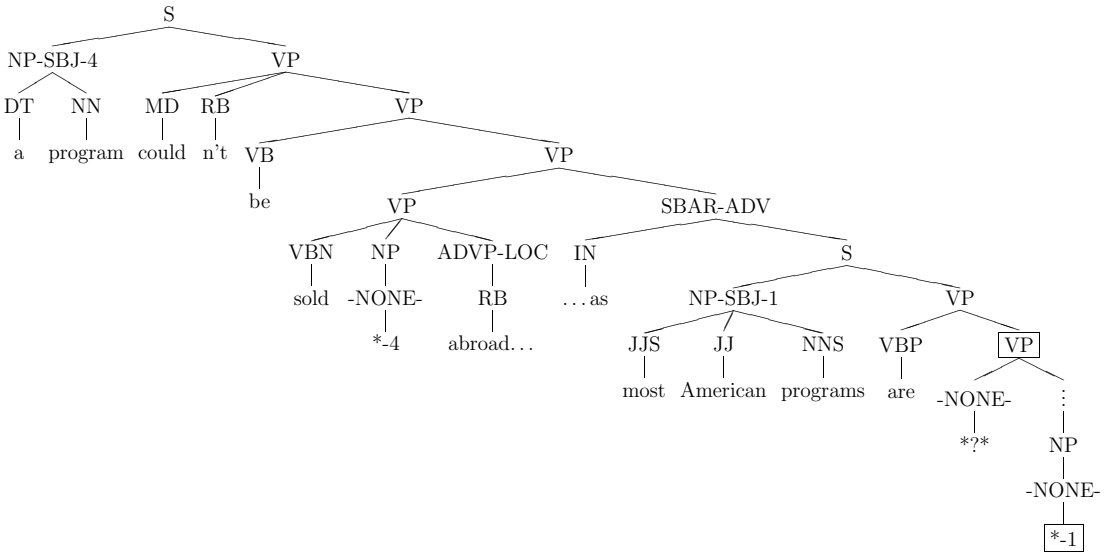

Figure 4.11: VPE parse missed by original empty VP feature

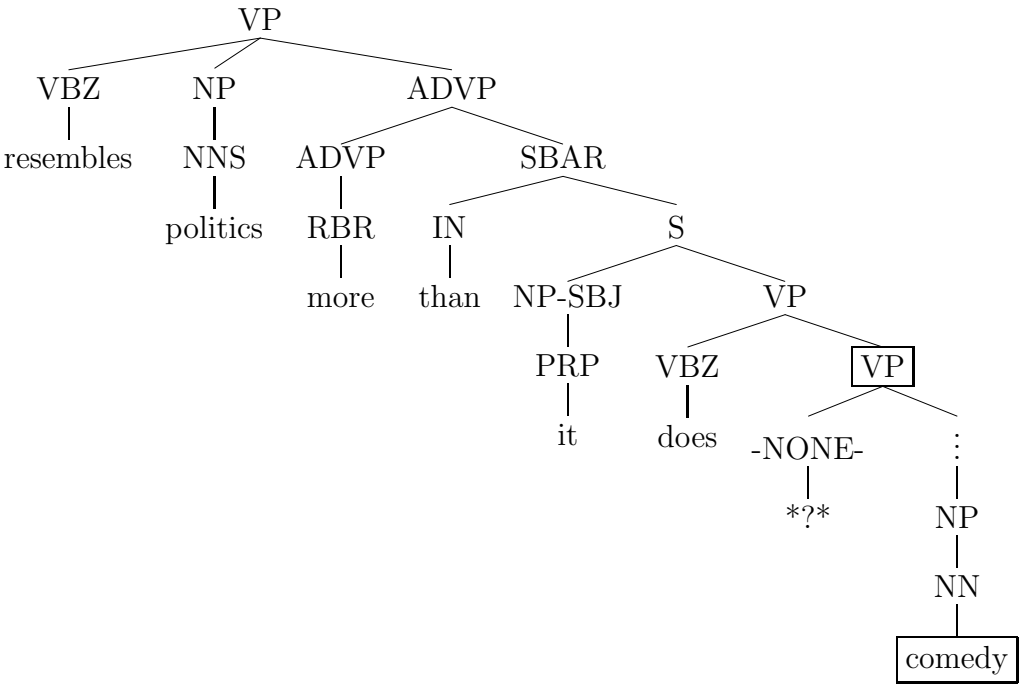

Figure 4.12: Pseudo-gapping parse missed by original empty VP feature

Algorithm	Recall	Precision	F1	|ER|	%ER	Significance
Empty VP	43.13	98.50	60.00			
MBL	68.62	73.94	71.18	4.06	12.35	none
GIS-MaxEnt	64.70	85.34	73.60	9.73	26.93	0.001
L-BFGS-MaxEnt	73.85	83.08	78.20	3.8	14.84	none
SLIPPER	75.82	36.25	49.05	1.21	2.32	0.05

Table 4.31: Effects of using the Empty VP feature

contains an empty element, but can contain other categories as well, gives the results seen in Table 4.32. These results are compared to the original empty-VP feature. This improves the feature itself by 10% in F1, and gives between 4.5% and 11% |ER| to the classifiers, halving L-BFGS-MaxEnt's error rate.

The original version of the feature shows statistical significance for two of the classifiers, and the improved version for all classifiers. When compared to results without any version of the feature, all classifiers show statistical significance of 0.001.

Algorithm	Recall	Precision	F1	|ER|	%ER	Significance
Empty VP	54.90	97.67	70.29			
MBL	77.12	77.63	77.37	6.19	21.48	0.001
GIS-MaxEnt	69.93	88.42	78.10	4.5	17.05	0.05
L-BFGS-MaxEnt	83.00	90.07	86.39	11.19	51.33	0.001
SLIPPER	86.27	39.17	53.88	4.83	9.48	0.001

Table 4.32: Effects of using the improved Empty VP feature

While it is clear that the empty VPs encoded in the Treebank identify VPE, we could not discern a pattern in those that it misses, which are for the most part unambiguous examples of VPE. The parsing guide provided with the Treebank states that

> policy for *?* was never finalized, so its use varies to some extent. In general, *?* is used by the annotators as a last resort (short of the FRAG analysis) for the annotation of clauses with "missing" material.

which may explain why it does not cover all cases of VPE.

4.3.9 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 4.33 shows that adding this information results in 2-4% |ER| for MBL and the MaxEnt classifiers.

Algorithm	Recall	Precision	F1	|ER|	%ER	Significance
MBL	83.00	79.87	81.41	4.04	17.85	none
GIS-MaxEnt	75.16	90.55	82.14	4.04	18.45	0.05
L-BFGS-MaxEnt	86.27	90.41	88.29	1.9	13.96	none
SLIPPER	82.35	62.07	70.79	16.91	36.67	0.001

Table 4.33: Effects of using the empty categories

For SLIPPER, the increase is much greater, and the rules learned show why (Figure 4.13). The first two rules are to be expected, but rule 3 is interesting. It states that if the next leaf in the tree is an elided material (*?*), and the current word is a form of the word 'be', it must be elliptical. This should in fact be covered by the Empty VP rule, but indicates that there are cases where although the fact that elided material exists is marked as such by the Treebank, they are not marked as belonging to a VP, as they should be.

Rule 4 is used to catch cases missed by Auxiliary-final VP and the Heuristic baseline, where a possible VPE is preceded by a comma. This is, in effect, the close to punctuation feature, but perhaps a more reliable one, suggesting a possible improvement.

Rank	VPE if
1	Empty VP = true
2	Auxiliary-final VP = true
3	word[+1]=*?* and word=VBX(a form of 'be')
4	Previous category = , and word=VPX(auxiliary or modal) and Auxiliary-final VP = false and Heuristic baseline = false
5	word=VDX(a form of 'do')
6	word=VPX(auxiliary or modal)

Figure 4.13: Rules learned by SLIPPER with features

4.3.10 Using extracted features only

We have seen that the addition of the features gives marked improvements over just words and POS tags, but given the high performance of some of these features, the question can be asked whether the word and POS tag information is needed in the classification process. Table 4.34 shows the results for an experiment using only the extracted features. It is seen that these results are only marginally better than the Empty VP feature by itself, and fall short of results when using the context provided by the words and POS tags too. It is interesting to note that when features with a large number of classes (i.e. words and POS tags) are removed, MBL performs better than the rest, albeit by a very small margin.

Algorithm	Recall	Precision	F1
MBL	68.83	79.70	73.87
GIS-MaxEnt	71.43	73.83	72.61
L-BFGS-MaxEnt	73.38	73.38	73.38
SLIPPER	43.51	94.37	59.56

Table 4.34: Performance using only extracted features

4.3.11 Voting

A simple way of combining the information from the different classifiers is a voting scheme. The most common form is majority voting, where a majority of the classifiers need to agree for a result to be accepted. This can lead to improvements in results, as spurious errors not found in the majority of the classifiers will be eliminated. Table 4.35 shows results for four levels of voting, ranging from disjunction to conjunction of the outputs of the four classifiers. The row for One-vote shows results where any classifier voting positively gives a positive result, and the row for Four-vote shows results where all classifiers must vote positively for a positive result. Accepting a potential VPE when at least two classifiers vote positively gives the best results, but this is only marginally (0.09%) better than the results for L-BFGS-MaxEnt alone.

Interestingly, precision does not increase linearly as more votes are needed. Two-vote has 87% precision, and it would be expected for Three-vote to have higher

Votes	Recall	Precision	F1
One (disjunction)	92.81	57.48	71.00
Two	89.54	87.26	88.38
Three (majority)	79.08	85.81	82.31
Four (conjunction)	65.35	83.33	73.26

Table 4.35: Voting scheme

precision than this, as one more vote is needed for positive results; but this is not the case. This is due to 20 incorrect votes common to all classifiers; because precision is tied to correct guesses as well as incorrect guesses, the higher votes get reduced precision as well as recall.

The maximum entropy algorithms have 90% precision, but MBL and SLIPPER score lower. It is possible that the low precision of these algorithms aversely affects the voting procedure. To check for this case, the effects of maximising precision was investigated, optimising all algorithms but MBL for this, which cannot be weighted that way. Highest precision is achieved using a confidence threshold of 0.7 for GIS-MaxEnt, 0.55 for L-BFGS-MaxEnt, and no weighting for SLIPPER[8]. The results of this approach, as well as the performance of the modified algorithms is seen in Table 4.36.

This method sacrifices too much recall and the overall effect is a lower score. The results indicate that while it may be possible to achieve some benefit from the voting scheme, it is not enough to warrant further experimentation.

Votes	Recall	Precision	F1
GIS-MaxEnt	40.52	98.41	57.40
L-BFGS-MaxEnt	80.39	92.48	86.01
SLIPPER	47.05	90.00	61.80
One (disjunction)	91.50	77.34	83.83
Two	77.12	94.40	84.89
Three (majority)	50.32	91.66	64.97
Four (conjunction)	32.02	87.50	46.88

Table 4.36: Voting scheme - precision optimised

[8]These experiments also show that the F1-optimal setting is now 0.1 for GIS-MaxEnt, giving 84.76%. For SLIPPER, a weighting of 2 gives 71.69% F1. L-BFGS-MaxEnt is still optimal at 0.35. These new settings will not be adopted for reasons of comparability.

4.3.12 Stacking

For this experiment, L-BFGS-MaxEnt will be used as the final classifier, and the rest as the base classifiers (as described in section 2.2.6). This is because L-BFGS-MaxEnt has the highest performance among the algorithms, and it would be possible for the others to simply default to it. The test data will remain as before, and the training data will be split into two, with a balance of Wall Street Journal and Brown sections to each[9].

Table 4.37 shows the results of this experiment. The first three rows show the performance of the base learners trained using level zero training data on the test data. The final two lines are the performance of L-BFGS-MaxEnt containing all the normal features plus the predictions of the base classifiers.

When using the predictions from the base classifiers, it is possible to use them as binary (true/false) decisions or continuous probability distributions. The first of the final two lines of the table incorporates the GIS-MaxEnt predictions as binary values, while the second contains their probability distribution. Using continuous values for GIS-MaxEnt gives worse results than binary values. Experiments using continuous values for MBL and SLIPPER were not performed due to the lower scores obtained using the continuous GIS-MaxEnt data.

Stacking gives lower results than the best results obtained using L-BFGS-MaxEnt by itself (Table 4.33), which is likely to be due to insufficient training data.

Classifier	Recall	Precision	F1
MBL	70.53	77.80	73.99
GIS-MaxEnt	63.07	87.60	73.34
SLIPPER	51.24	94.64	66.49
L-BFGS-MaxEnt (binary GIS-MaxEnt)	86.92	83.64	85.25
L-BFGS-MaxEnt (continuous GIS-MaxEnt)	89.54	76.11	82.28

Table 4.37: Stacking scheme

[9]Level zero : WSJ 00, 01, 03, Brown CF, CG, CM.
Level one : WSJ 04, 15, Brown CL,CN,CP.

4.3.13 Gain ratio of features

In the experiments described, features were added one by one, and a comprehensive full permutation of the feature combinations was not carried out. Such a table might have given more insight into how the features interact, and how much they contribute to the overall performance. However, given the number of features this would have proven time consuming, and perhaps also difficult to interpret due to the large number of results which would be obtained.

TiMBL provides several measures of the contribution of the features, and by default, uses gain ratio divided by the number of values the feature has. Looking at the results for the Treebank data (see Appendix D.1 for the full table, including measures other than gain ratio), we get the ordering of the features in Table 4.38.

Rank	Feature	Values	Gain Ratio
1	*Empty VP*	2	0.17979602
2	*Auxiliary-final VP*	2	0.042644430
3	*Heuristic*	2	0.033044246
4	*Close to punctuation*	2	0.00053813059
5	*Tag*	8	0.00048445462
6	Tag_{+1}	50	0.00035095910
7	Tag_{+2}	49	0.00019923309
8	Tag_{-1}	46	0.00017237752
9	Tag_{-2}	49	0.00015724614
10	Tag_{+3}	50	0.00015842766
11	Tag_{-3}	50	0.00009473151
12	*Next category*	146	0.00015736589
13	*Previous category*	386	0.00021347748
14	$Word_{+1}$	10315	0.00058360551
15	$Word_{-1}$	9354	0.00040863022
16	$Word_{-3}$	12137	0.00043919421
17	*Word*	8103	0.00027172178
18	$Word_{-2}$	11192	0.00032733878
19	$Word_{+3}$	13210	0.00027832239
20	$Word_{+2}$	13194	0.00027148325

Table 4.38: Contribution of features

The top 3 ranks are as expected, given the high performance of these features by themselves. Close to punctuation gets to number 4, due to its binary outcome. Words, despite higher gain ratios on average than tags, are the least useful, as they have a large number of values. The surrounding categories are less useful

than any of the tags, as they have a larger distribution, given that they consist of category headers.

Among tags, the current tag is the most informative, followed by the next two tags. Among words, on the other hand, the next is most informative, followed by the previous.

4.3.14 Cross-validation

Cross-validation is performed with and without the features developed to measure the improvement obtained through their use (Table 4.39). The results for just words and POS tags are consistent with the development experiments. When the features are added, for MBL there is a 2.75% drop compared to results on held-out data experiments (Table 4.33), for GIS-MaxEnt 3%, for L-BFGS-MaxEnt 5.7%, and for SLIPPER 6.2%, which is not a large discrepancy, but does suggest that a level of overfitting exists.

On words and POS tags alone, GIS-MaxEnt and SLIPPER do rather poorly, while MBL and L-BFGS-MaxEnt get comparatively good results. Compared to the results obtained on the BNC, however, they are all lower.

With the addition of the features, it is GIS-MaxEnt and SLIPPER that benefit the most : over 40% improvement to F1. When seen in terms of %ER, all algorithms gain similar results, ranging between 54% and 68%. At this stage, all the algorithms get rather better results than those for the BNC. The results on the two corpora are not really comparable, but this is an issue that will be partially addressed in the next section.

Looking at the recall and precision figures, it is seen that MBL produces balanced results, L-BFGS-MaxEnt is more biased towards precision, and GIS-MaxEnt even more so, while SLIPPER is biased towards recall. Of course, the recall/precision balances of the maximum entropy algorithms depend on the threshold settings, while the balance for SLIPPER is dependent on the weighting used. With the features added the 10 times weighting is seen to be higher than optimal, but will be retained for further experiments.

	Words + POS			+ features				
Algorithm	Recall	Precision	F1	Recall	Precision	F1	|ER|	%ER
MBL	44.61	49.50	46.92	77.54	79.81	78.66	31.74	59.80
GIS-MaxEnt	21.55	76.59	33.64	71.10	89.11	79.10	45.46	68.51
L-BFGS-MaxEnt	54.19	73.42	62.36	78.74	86.94	82.63	20.27	53.85
SLIPPER	18.56	30.31	23.02	76.19	56.11	64.63	41.61	54.05

Table 4.39: Cross-validation on the Treebank

4.4 Experiments with Automatically Parsed data

The next set of experiments use the BNC and Treebank, but strips POS and parse information, and parses them automatically using two different parsers. This is both to overcome training data limitations, making the system more robust, and also to enable it to work on unannotated text.

4.4.1 Parsers used

Charniak's parser (2000) is a combination probabilistic context free grammar and maximum entropy parser[10]. It achieves a 90.1% recall and precision average for sentences of 40 words or less, and 89.5% for sentences of 100 words or less, on sections of the Penn Treebank.

Collins' parser (1999), like Charniak's, is trained on the Penn Treebank, and has similar performance[11]. Due to these similarities, it is expected that it would give similar results if used for the task at hand, and so I did not use it. Preiss (2003) shows that for the task of anaphora resolution, these two parsers, and even an earlier version of RASP, produce very similar results.

Robust Accurate Statistical Parsing (RASP) (Briscoe and Carroll, 2002) uses a combination of statistical techniques and a hand-crafted grammar[12]. RASP was chosen due to the fact that it is trained on a range of corpora, and that it uses a more complex tagging system (CLAWS-2), like that of the BNC[13]. This parser, on the datasets used, generated full parses for 70% of the sentences, partial parses

[10]Available from ftp://ftp.cs.brown.edu/pub/nlparser/
[11]Available from http://www.ai.mit.edu/people/mcollins/code.html
[12]Available from http://www.informatics.susx.ac.uk/research/nlp/rasp/
[13]For details see Appendix C.

for 28%, while 2% were not parsed, returning POS tags only.

4.4.2 Empty category information

While Charniak's parser does not generate empty-category information, Johnson (2002) has developed a pattern matching algorithm trained on the Treebank that can insert empty categories into the parser's output[14]. The algorithm achieves 79% F1 when used on parsed data (section 23 of the WSJ), but it must be noted that this is an aggregate score for all the empty categories, and empty VPs occur too rarely to be included in the score tables for the most common categories. This program is used in conjunction with Charniak's parser.

More recently Dienes and Duby (2003a; 2003b) propose a method where empty categories are inserted in POS-tagged text, and then a PCFG parser is run on the output of this step which finds non-local dependencies, getting 74.6% F1 for the same task. It must be noted that in addition to inserting the empty categories, they also resolve trace antecedents (which for Johnson's algorithm is at 68% accuracy). Campbell (2004) opts for a non-statistical, rule based approach that achieves 83.4% F1. Jijkoun and de Rijke (2004) present an approach based on graph rewriting using memory based learning, which is applied to dependency structures, and not phrase trees, as is the case with the other approaches discussed. The authors of this work do not present an exact comparison to the other algorithms, but get results comparable to the Dienes and Duby approach.

Unfortunately, none of the algorithms except Johnson's were available for use when experiments were started, so empty category information is available only for Charniak's parser and not RASP.

4.4.3 Reparsing the Treebank

Figure 4.14 summarises the results for experiments done using the held-out division of data[15], and Table 4.41 using cross-validation, on data from the Treebank parsed by Charniak's parser.

[14] Available from http://www.cog.brown.edu/∼mj/Software.htm
[15] Detailed tables and figures can be found in Appendix D.2

Without the features, the results are similar to those on the manually annotated Treebank, especially using cross-validation. This is to be expected, as the Treebank and Charniak's parser use the same tagging schema, and the results suggest that the POS tagging part is working well. With the features, the results are between 15 and 22% F1 lower.

Seen in %ER, the effect of the features is now 14%-32%, around half of what was achieved on the Treebank. The reason for this can be seen in the success rate of the features as seen independently of the classifiers, with the results in Table 4.40 showing the performance of the heuristics themselves. The close-to-punct feature shows reduced performance, the heuristic baseline stays the same, and the auxiliary-final VP shows a 5% drop. The empty VP feature retains a high precision of over 80%, but its recall drops by 50% to 20%. Empty VP still gives consistent improvements (except for SLIPPER, which does not use it in any rule), but the addition of empty categories in general reduces performance for three of the algorithms. While statistics for the success rate of these other empty categories was not collected, it can be surmised from the results that they are degraded to a large extent.

Feature	Recall	Precision	F1
close-to-punct	34.00	2.47	4.61
heuristic baseline	45.33	25.27	32.45
auxiliary-final VP	51.33	36.66	42.77
empty VP	20.00	83.33	32.25

Table 4.40: Performance of features on Charniak parsed Treebank data

	Words + POS			+ features				
Algorithm	Recall	Precision	F1	Recall	Precision	F1	|ER|	%ER
MBL	45.01	47.45	46.20	59.51	63.04	61.22	15.02	27.92
GIS-MaxEnt	23.86	77.07	36.44	46.97	71.99	56.85	20.41	32.11
L-BFGS-MaxEnt	53.62	73.80	62.11	62.83	73.11	67.58	5.47	14.44
SLIPPER	21.14	22.15	21.63	63.44	33.70	44.02	22.39	28.57

Table 4.41: Cross-validation on the Charniak parsed Treebank

Experiments on the RASP-parsed Treebank data give the results in Figure 4.15, and cross-validation results are seen in Table 4.43.

Compared to the results for Charniak's parser, the results are similar without features, except MBL which does 11% better, perhaps due to the more fine-grained

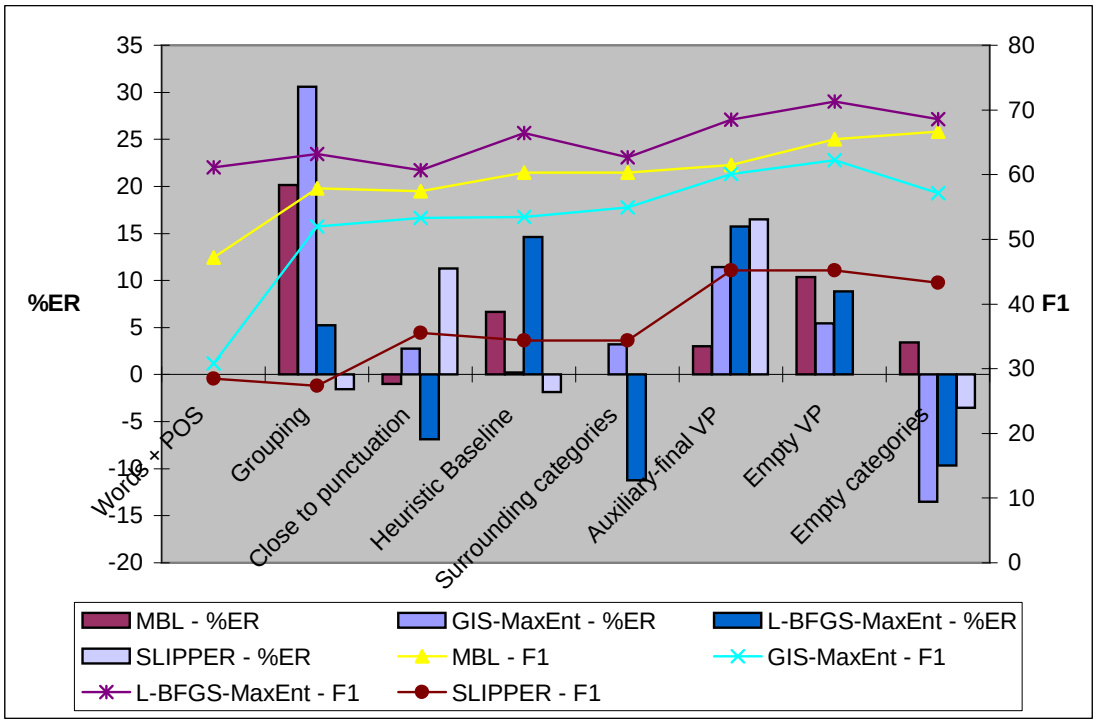

Figure 4.14: F1 and Percentage Error Reduction plots for classifiers on Charniak parsed Treebank data versus features being added

POS tagging employed. Using the features, MBL does 5% better and GIS-MaxEnt 6%, while L-BFGS-MaxEnt and SLIPPER show similar results.

RASP performs lemmatization on the data, and using the lemmas can act as a form of smoothing, but it gets mixed results, improving performance for GIS-MaxEnt but reducing it for the other two, with no effect on SLIPPER. The features involving empty-category information cannot be replicated, as RASP does not generate these.

Looking at the success rate of the features used (Table 4.42), it is seen that performance for close-to-punct and the heuristic baseline stays the same. Auxiliary-final VP, which is determined by parse structure, is only half as successful, and while giving improvement for L-BFGS-MaxEnt, reduces performance for MBL and GIS-MaxEnt. The heuristic baseline, which depends on POS information, gives better results for RASP. The %ER of the features in total is between 13%-44%.

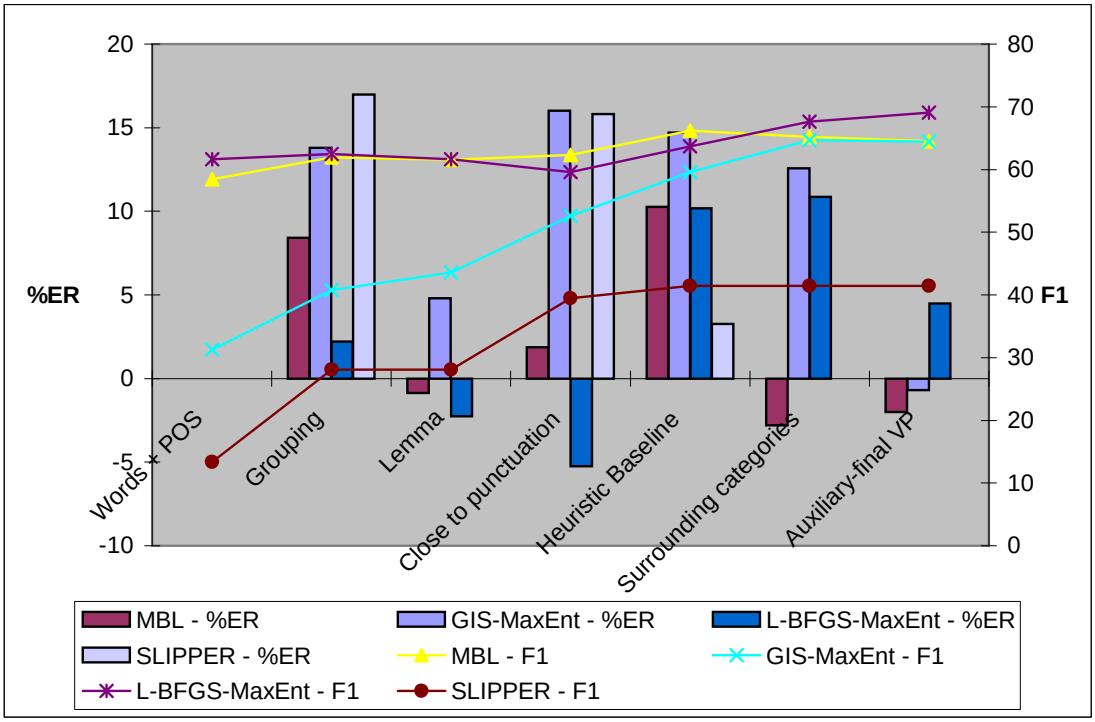

Figure 4.15: F1 and Percentage Error Reduction plots for classifiers on RASP parsed Treebank data versus features being added

Feature	Recall	Precision	F1
close-to-punct	71.05	2.67	5.16
heuristic baseline	74.34	28.25	40.94
auxiliary-final VP	22.36	25.18	23.69

Table 4.42: Performance of features on RASP parsed Treebank data

	Words + POS			+ features				
Algorithm	Recall	Precision	F1	Recall	Precision	F1	|ER|	%ER
MBL	52.99	63.10	57.60	62.27	70.74	66.24	8.64	20.38
GIS-MaxEnt	22.90	59.76	33.10	55.53	73.03	63.09	29.99	44.83
L-BFGS-MaxEnt	53.29	74.16	62.02	63.47	71.50	67.24	5.22	13.74
SLIPPER	15.41	19.92	17.38	76.79	30.08	43.23	25.85	31.29

Table 4.43: Cross-validation on the RASP parsed Treebank

4.4.4 Parsing the BNC

For comparison purposes experiments were performed using a parsed version of the BNC corpora. Figure 4.16, and Tables 4.45 and 4.44 summarise results using the Charniak parser[16], while Figure 4.17, and Tables 4.47 and 4.46 summarise results using RASP.

Compared to the original results, MBL does 4.5% worse using the Charniak parser and 6% worse using RASP. GIS-MaxEnt does 5.5% and 2.5% better, respectively, while L-BFGS-MaxEnt does 4% and 6.7% worse. SLIPPER again trails behind in performance, and Charniak's parser gives consistently better results. This is not really due to any improvement from the empty category information, as most of the classifiers hit their peak performance without this information for Charniak's parser. The empty VP feature suffers from further reduced performance for the BNC data, which it is not trained on, with another 7% drop.

The addition of features gives around 30% %ER most of the time, except for L-BFGS-MaxEnt, which only derives small improvements from them, 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.

Feature	Recall	Precision	F1
close-to-punct	48.00	5.52	9.90
heuristic baseline	44.00	34.50	38.68
auxiliary-final VP	53.00	42.91	47.42
empty VP	15.50	62.00	24.80

Table 4.44: Performance of features on Charniak parsed BNC data

	Words + POS			+ features				
Algorithm	Recall	Precision	F1	Recall	Precision	F1	|ER|	%ER
MBL	55.75	54.13	54.93	68.82	66.05	67.41	12.48	27.69
GIS-MaxEnt	37.88	75.96	50.56	61.51	72.05	66.36	15.8	31.96
L-BFGS-MaxEnt	63.54	75.71	69.10	70.50	73.04	71.75	2.65	8.58
SLIPPER	35.49	26.95	30.64	77.45	43.35	55.59	24.95	35.97

Table 4.45: Cross-validation on the Charniak parsed BNC

[16]Detailed tables and figures for this section can be found in Appendix D.3

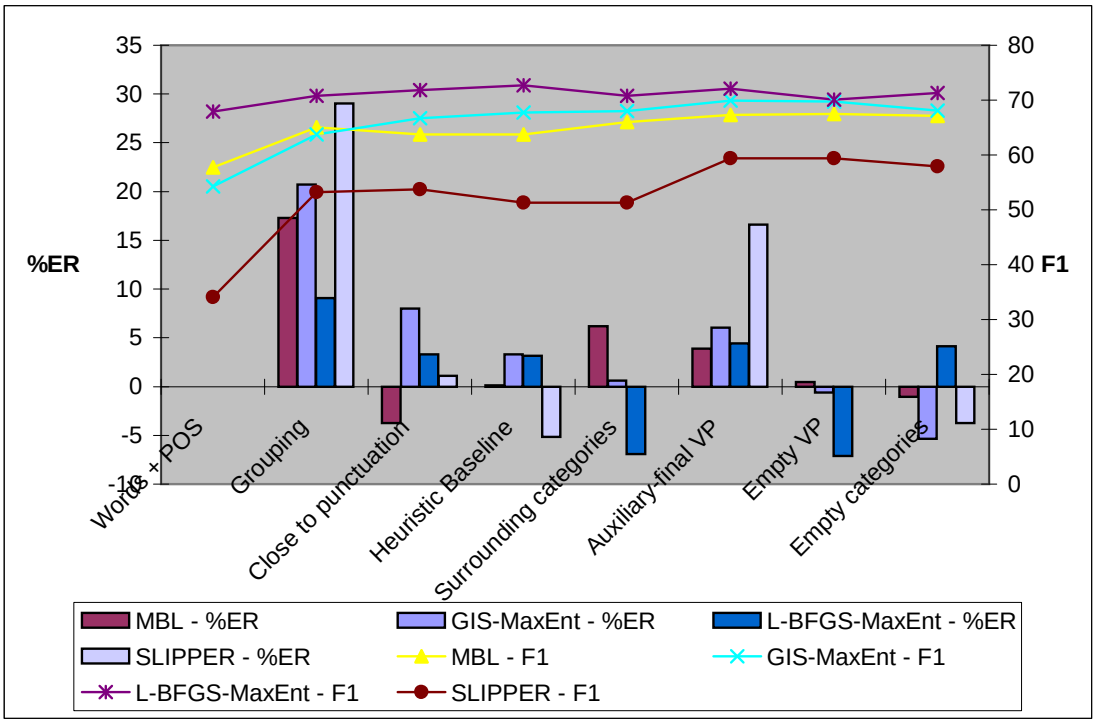

Figure 4.16: F1 and Percentage Error Reduction plots for classifiers on Charniak parsed BNC data versus features being added

Feature	Recall	Precision	F1
close-to-punct	55.32	4.06	7.57
heuristic baseline	84.77	35.15	49.70
auxiliary-final VP	16.24	28.57	20.71

Table 4.46: Performance of features on RASP parsed BNC data

	Words + POS			+ features				
Algorithm	Recall	Precision	F1	Recall	Precision	F1	|ER|	%ER
MBL	56.51	63.97	60.01	64.99	67.18	66.07	6.06	15.15
GIS-MaxEnt	39.57	63.50	48.76	65.49	70.16	67.74	18.98	37.04
L-BFGS-MaxEnt	60.22	73.50	66.20	67.50	71.16	69.28	3.08	9.11
SLIPPER	21.07	11.84	15.16	79.92	37.80	51.32	36.16	42.62

Table 4.47: Cross-validation on the RASP parsed BNC

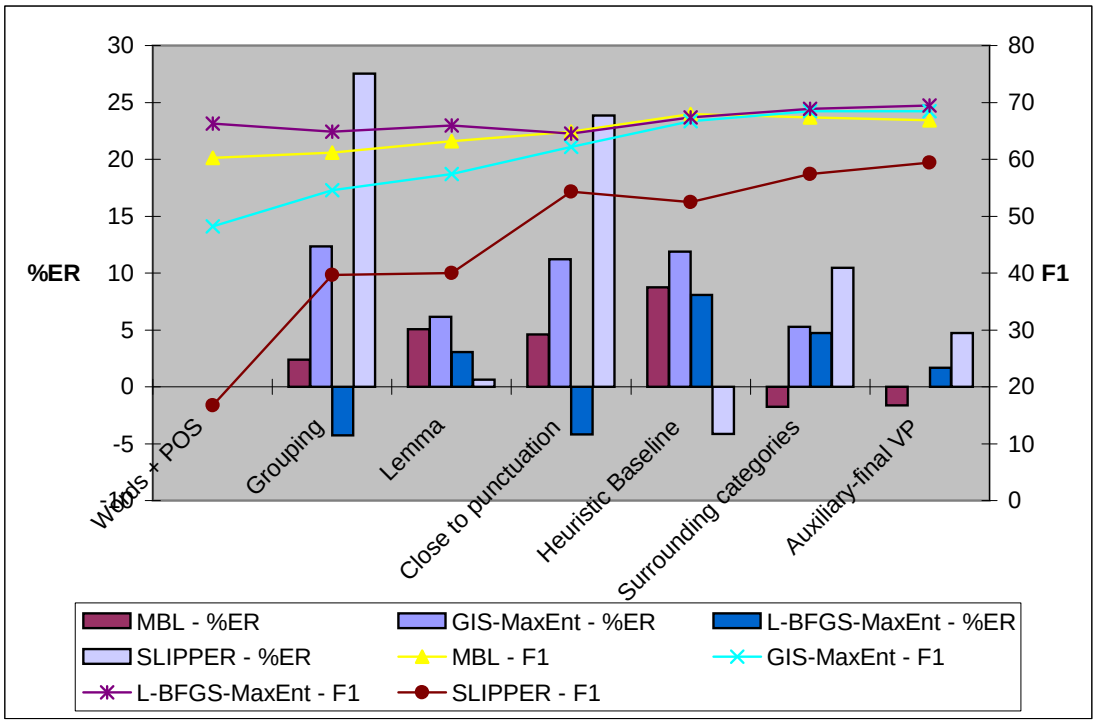

Figure 4.17: F1 and Percentage Error Reduction plots for classifiers on RASP parsed BNC data versus features being added

4.4.5 Combining BNC and Treebank data

Combining the RASP-parsed BNC and Treebank data diversifies and increases the size of the training and test data, which should serve both to make the conclusions drawn more reliable, and the classifiers constructed more robust. Figure 4.18 and Table 4.48 summarise results using the Charniak parser[17], while Figure 4.19 and Table 4.49 summarise results using RASP.

The results for this experiment are not very different from those for the individual datasets, but the results are generally an improvement over the weighted average of the individual results (Table 4.50). The differences are not very large, which 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 may also limit the relevance of examples from one to the other. It may also be noted that for SLIPPER the increase in data size seems to be a detriment instead of an improvement.

[17]For complete tables and figures for this section see Appendix D.4

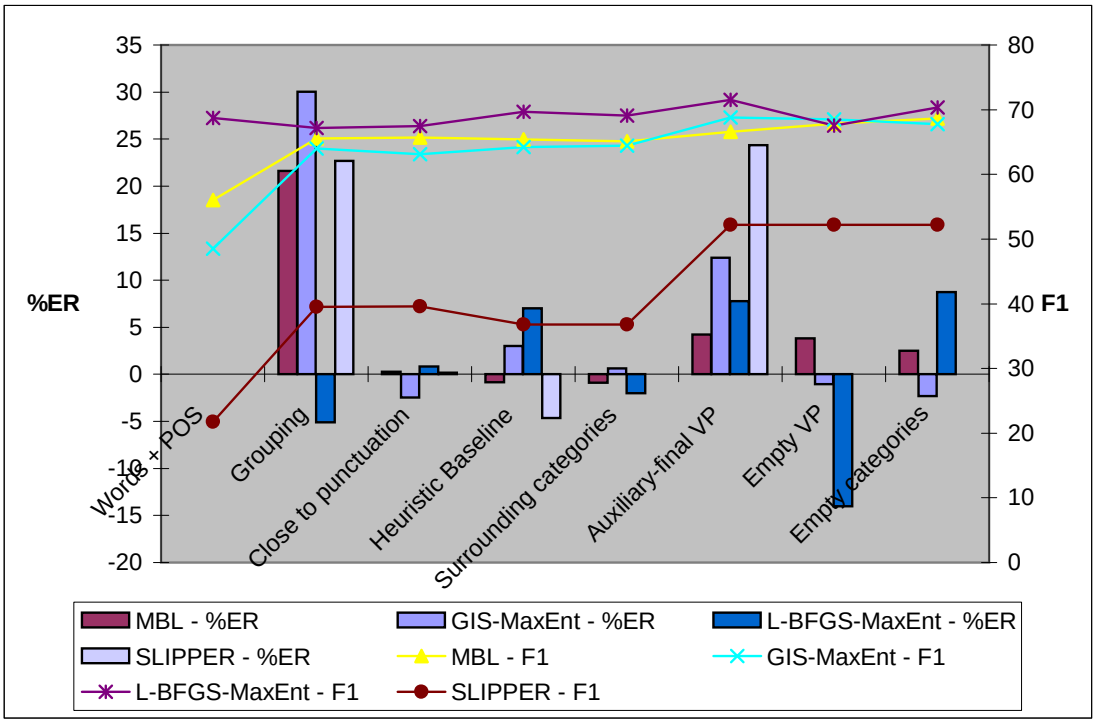

Figure 4.18: F1 and Percentage Error Reduction plots for classifiers on Charniak parsed combined data versus features being added

	Words + POS			+ features				
Algorithm	Recall	Precision	F1	Recall	Precision	F1	|ER|	%ER
MBL	52.740	55.05	53.87	66.51	68.15	67.32	13.45	29.16
GIS-MaxEnt	37.83	78.72	51.10	61.43	73.93	67.10	16.00	32.72
L-BFGS-MaxEnt	63.97	72.72	68.06	68.44	72.16	70.25	2.19	6.86
SLIPPER	29.27	20.26	23.95	73.12	37.77	49.81	25.86	34.00

Table 4.48: Cross-validation on the Charniak parsed combined dataset

	Words + POS			+ features				
Algorithm	Recall	Precision	F1	Recall	Precision	F1	|ER|	%ER
MBL	58.29	64.40	61.19	64.84	69.95	67.30	6.11	15.74
GIS-MaxEnt	37.26	65.31	47.45	62.32	71.05	66.40	18.95	36.06
L-BFGS-MaxEnt	62.38	72.94	67.25	67.91	72.31	70.04	2.79	8.52
SLIPPER	19.11	11.64	14.47	80.40	31.23	44.98	30.51	35.67

Table 4.49: Cross-validation on the RASP parsed combined dataset

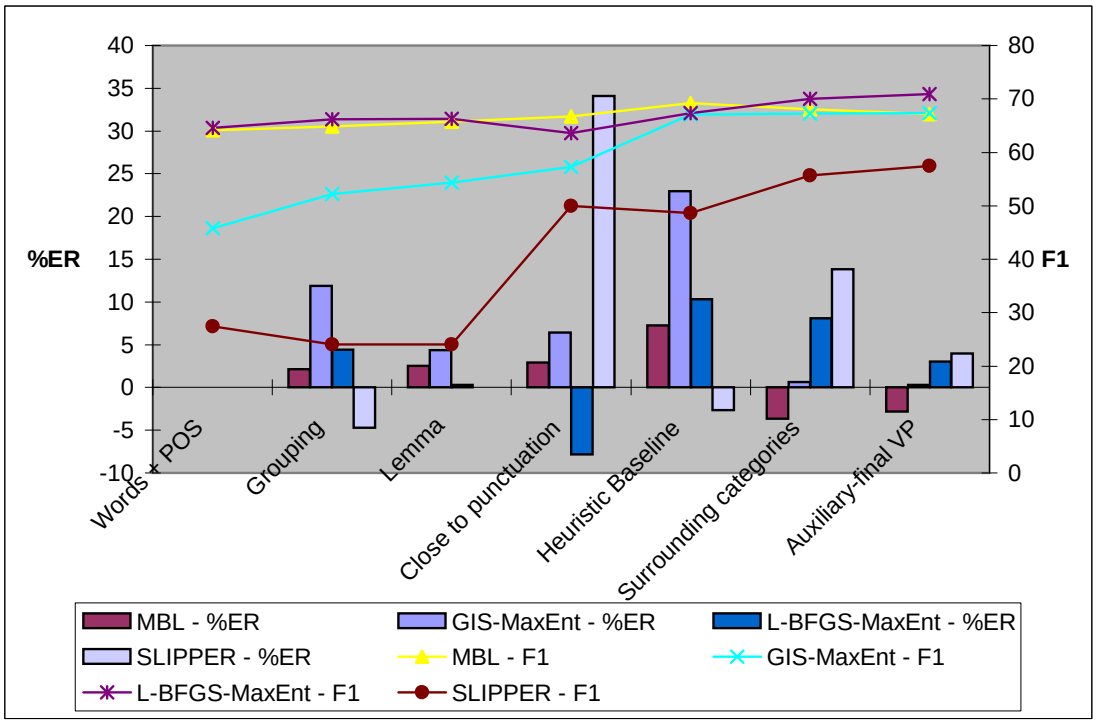

Figure 4.19: F1 and Percentage Error Reduction plots for classifiers on RASP parsed combined data versus features being added

	Charniak			RASP		
Algorithm	Average	Combined	Difference	Average	Combined	Difference
MBL	64.73	67.32	2.59	66.14	67.30	1.16
GIS-MaxEnt	62.25	67.10	4.85	65.73	66.40	0.67
L-BFGS-MaxEnt	69.95	70.25	0.30	68.40	70.04	1.64
SLIPPER	50.59	49.81	-0.78	47.82	44.98	-2.84

Table 4.50: Improvement over weighted average in F1

4.5 Error analysis

This section will assess categories of errors encountered in the experiments, and if it is possible to correct them, attempt to do so. The errors being examined will be those that are common to at least three of the classifiers, with the assumption that these are the ones that point out deficiencies in the approach or features, while those that do not generate this level of agreement are due to problems in the classifiers themselves. The analysis will be done for the Treebank data, and then for the combined data as parsed by the parsers. Due to the small number of

errors none of the improvements made in this section result in large changes, and none that are statistically significant (with the exception of SLIPPER at times, which again can be unpredictable).

4.5.1 Treebank data

Other empty phrases The test data contains forms of elided material which are not interpreted as belonging to a VP, as seen in Figures 4.20 (3 cases) and 4.21 (2 cases).

Figure 4.21 is a case of pseudo-gapping, which may explain why the empty VP feature as it is does not catch it. On the other hand, Figure 4.20 is a VPE, with the antecedent being 'mad'. The (ADJP-PRD (-NONE- *?*)) and (NP-PRD (-NONE- *?*)) constructions occur in non-elided contexts as well, so they are not necessarily meant to mark out VPEs. The results on the entire development set (Table 4.51) show that while adding the NP-PRD data improves performance, further adding ADJP-PRD doesn't, so only NP-PRD will be used. The effect of the change on classification accuracy itself (Table 4.52) is small.

Algorithm	Recall	Precision	F1
Empty VP	54.90	97.67	70.29
Empty VP & ADJP-PRD	58.82	88.23	70.58
Empty VP & NP-PRD	58.16	96.73	72.65
Empty VP & NP-PRD & ADJP-PRD	61.43	87.03	72.03

Table 4.51: Empty VP performance using other empty phrases

Algorithm	Recall	Precision	F1	|ER|	%ER
MBL	82.35	80.76	81.55	1.16	5.92
GIS-MaxEnt	77.77	90.83	83.80	1.66	9.29
L-BFGS-MaxEnt	88.23	90.00	89.10	0.81	6.92
SLIPPER	48.36	88.09	62.44	-8.35	-28.59

Table 4.52: Effects of using other empty phrases

Empty VP conflicting with other data There are 5 cases where the Empty VP feature predicting no VPE conflicts with the correct VPE prediction of one or more of the other features, and there is 1 case where the correct VPE prediction

```
( (S (`` ``)
    (S-TPC-1
      (NP-SBJ (PRP I) )
      (VP (VBP 'm)
        (ADJP-PRD (JJ mad) )))
    ('' '')
    (, ,)
    (SINV
      (VP (VBD shouted)
        (S (-NONE- *T*-1) ))
      (NP-SBJ (NNP Payne) )
      (, ,)
      (SBAR-ADV (IN as)
        (S
          (NP-SBJ (PRP he) )
          (VP (VBD ran)
            (PP-DIR (IN out)
              (PP (IN into)
                (NP (DT the) (NN hall) )))))))
    (. .) ))

( (S (`` ``)
    (S
      (NP-SBJ (PRP I) )
      (VP (VBP 'm)
        (ADJP-PRD (JJ mad) )))
    ('' '')
    (, ,)

    (CC and)
    (S
      (NP-SBJ (-NONE- *) )
      (VP (RB only) (VBD wished)
        (SBAR (-NONE- 0)
          (S
            (NP-SBJ (PRP he) )
            (VP (VBD had)
              (VP (VBN been)
                (ADJP-PRD (-NONE- *?*) )))))))
    (. .) ))
```

Figure 4.20: Empty ADJP VPE parse

```
( (S
    (NP-SBJ (PRP She) )
    (VP (VBD was)
      (NP-PRD (DT the) (NN pursuer) )
      (ADVP
        (ADVP (RB as) (RB clearly) )
        (SBAR (IN as)
          (SINV (VBD was)
            (NP-SBJ (NNP Venus) )
            (NP-PRD (-NONE- *?*) )
            (PP (IN in)
              (NP
                (NP (NNP Shakespeare) (POS 's) )
                (NN poem) ))))))
    (. .) ))
```

Figure 4.21: Empty NP VPE parse

of the empty VP feature is overturned by the other features. It may therefore be useful to modularise empty VP as a separate step, and run the test on samples it does not identify as VPE. Doing this gives the results in Table 4.53, where the first row for each classifier gives results for the remaining test set, and the second row the combined results.

When compared to the previous results (those in the previous section, with the expanded empty VP), all classifiers get improved performance. This is a side effect of the design principles of the classifiers used, where a single weight is set for positive and negative predictions from a feature, which means that it cannot take into account that a feature may reliably predict one output, but not the other. A decision tree approach would be able to incorporate it more successfully, effectively reproducing what we do here. The method used here cannot be applied to the automatically parsed data as the precision for the empty VP feature is too low.

Traces There are 5 cases in the test data where a trace is inserted instead of an elided material marker (Figure 4.22). There is no simple way to deal with this, as traces are very numerous and a feature to encompass them would suffer from very low precision.

Algorithm	Recall	Precision	F1	|ER|	%ER
MBL	70.76	56.79	63.01		
combined	87.66	77.58	82.31	0.76	4.12
GIS-MaxEnt	66.15	76.78	71.07		
combined	85.71	88.59	87.12	3.32	20.49
L-BFGS-MaxEnt	76.92	79.36	78.12		
combined	90.25	89.10	89.67	0.57	12.23
SLIPPER	7.69	41.66	12.98		
combined	61.03	89.52	72.58	10.14	27.00

Table 4.53: Performance with Empty VP feature used as a preprocessing step

```
...
        (VP
          (ADVP (RB probably) )
          (VBG charging)
          (ADVP (JJ double) )
          (SBAR-NOM
            (WHNP-2 (WP what) )
            (S
              (NP-SBJ (JJ ordinary) (NNS maids) )
              (VP (VBD did)
                (NP (-NONE- *T*-2) ))))
          (PP (IN for)
            (NP (NN housework) )))
                ...
```

Figure 4.22: Parse with trace

Comparatives There are 3 cases, such as the one in Figure 4.23, where comparative movement is mistaken for VPE. To correct this, the antecedent would have to be located first, which requires information not available at this stage. These cases exhibit very similar behaviour to VPE, and can be construed as VPE, so these cases are not clear errors, but they are not included in the experiments due to differences in their processing.

Main verb There are two cases where false positives are generated due to a main verb being identified as an auxiliary (Figure 4.24). This kind of case is difficult to distinguish from VPE. The fact that its object is a trace should prevent this. This is complicated by the fact that sometimes traces do get inserted in VPE sites, and sometimes they are left out after main verbs.

```
...
                (, ,)
                (NP-SBJ (PRP he) )
                (VP (MD would)
                  (VP (VB be)
                    (ADJP-PRD
                      (ADJP (RB far) (JJR wealthier) )
                      (SBAR (IN than)
                        (S
                          (NP-SBJ (PRP he) )
                          (VP (VBZ is)
                            (ADJP-PRD (-NONE- *?*) ))))))))))))
                  ...
```

Figure 4.23: Parse with comparative

```
( (SBARQ (`` ``)
    (WHNP-1 (WP What) (RB good) )
    (SQ (MD will)
      (NP-SBJ (PRP it) )
      (ADVP (RB really) )
      (VP (VB do)
        (NP (-NONE- *T*-1) )))
    ('' '') (. ?) (. ?) ))
```

Figure 4.24: Parse with main verb 'do'

Auxiliary-final question There are 3 cases where a VP is not explicitly marked inside an SQ header (Figure 4.25). SQ holds the inverted auxiliary (if there is one) and the rest of the sentence in *wh*-questions. A simple search for SQ's that only hold an auxiliary, and optionally negation or pronouns correctly identifies the 3 samples in the test corpus, with no false positives. Given its precision, we decided to use this check as a separate preprocessing step as well (Table 4.54). This feature serves as the question counterpart to the auxiliary-final VP feature.

Tagging/parsing error The parse in Figure 4.26 gives a false positive due to 'doubt' being mistagged :

```
( (SQ
    (S-IMP
      (NP-SBJ (-NONE- *) )
      (VP (VB 'fess)
        (PRT (RP up) )))
    (: --)
    (SQ (VBP do) (RB n't)
      (NP-SBJ (PRP you) ))
    (. ?) (. ?) ))
```

Figure 4.25: The SQ phrase header

Algorithm	Recall	Precision	F1	|ER|	%ER
MBL	69.84	56.41	62.41		
combined	87.66	78.03	82.56	0.25	1.41
GIS-MaxEnt	66.66	77.77	71.79		
combined	86.36	89.26	87.78	0.66	5.12
L-BFGS-MaxEnt	76.19	80.00	78.04		
combined	90.25	89.67	89.96	0.29	2.81
SLIPPER	9.52	50.00	16.00		
combined	62.98	90.65	74.32	1.74	6.35

Table 4.54: Performance with Auxiliary-final question feature used as a preprocessing step

```
( (S
    (NP-SBJ (PRP He) )
    (VP (VBD did) )
    (RB n't)
    (NP (NN doubt)
      (NP (PRP$ her) (NN truthfulness) )
      (, ,)
      (SBAR-ADV (IN although)
        (S
          (NP-SBJ (PRP he) )
          (VP (VBD had)
            (VP (VBN heard)
              (NP (DT the) (NNS words) )
              (NP-TMP (DT a) (CD hundred) (NNS times) ))))))
    (. .) ))
```

Figure 4.26: Mistagged parse

```
( (S
    (S
      (NP-SBJ (PRP He) )
      (VP (VBD was)
        (ADJP-PRD (JJ sure) )))
    (, ,) (IN for)
    (S
      (S
        (NP-SBJ (PRP he) )
        (VP (VBD had)
          (VP (VBN done)
            (SBAR-ADV (IN as)
              (S
                (NP-SBJ-1 (PRP he) )
                (VP (VBD was)
                  (VP (VBN told)
                    (NP (-NONE- *-1) ))))))))
      (, ,)
      (SQ
        (VP (VBD had) (RB n't) )
        (NP-SBJ (PRP he) )))
    (. ?) (. ?) ))
```

Figure 4.27: Parse where auxiliary VP is identified

Remaining cases The remaining 10 cases of false negatives seem simply to be too difficult for the classifiers given the limited context and the feature-set. The Close to punctuation and Heuristic baseline features are simplistic and do not perform well, but the Auxiliary-final VP and Empty VP features should, in an ideal corpus, be able to predict all VPE instances. Due to inconsistencies in the application of the parsing scheme, however, they don't. When one or more of the features do predict correctly, this may get overturned by one or more of the other features that don't, resulting in the cases for which there is no simple fix.

It is seen that the low success rate of the Auxiliary-final VP feature is due to inconsistencies in the application of the parse notation. In Figure 4.27, the VPE site is identified as having its own VP, while in Figure 4.28 it isn't. This also works the other way, where the feature is triggered incorrectly, resulting in 2 false positives.

```
( (S
    (NP-SBJ (NNS People) )
    (VP (VBD got)
      (ADJP-PRD (JJ rich) )
      (PP-MNR (IN through)
        (NP (NNS takeovers) ))
      (PP-TMP (IN in)
        (NP (DT those) (NNS days) ))
      (, ,)
      (SBAR-MNR (IN as)
        (S
          (NP-SBJ (PRP they) )
          (VP (VBP do)
            (NP-TMP (NN today) )))))
    (. .) ('' '') ))
```

Figure 4.28: Parse where auxiliary VP is not identified

Cross-validation Incorporating the two preprocessing stages discussed, the cross-validation results given in Table 4.55 are seen. The first row for each algorithm represents its performance on the data that has not been handled by the features used for preprocessing the data, and the second row shows the combined performance. It is seen that correcting for some simple systematic errors dependent on the corpus, it is possible to improve %ER by 10-20%.

Algorithm	Recall	Precision	F1	|ER|	%ER
MBL	63.52	60.00	61.71		
combined	83.26	78.45	80.78	2.12	9.93
GIS-MaxEnt	58.30	73.06	64.85		
combined	80.86	85.87	83.29	4.19	20.05
L-BFGS-MaxEnt	66.77	74.54	70.44		
combined	84.75	85.90	85.32	2.69	15.49
SLIPPER	13.68	50.60	21.54		
combined	60.39	86.32	71.06	6.43	18.18

Table 4.55: Cross-validation using preprocessed features on Treebank data

4.5.2 Charniak data

Auxiliary-final question The Charniak parser, having been trained on the Treebank, produces the SQ structure discussed before. Using the Auxiliary-final

question feature, 8 true positives and 8 false positives are returned. As the precision is too low to use this as a stand-alone classifier, it is added as a feature, giving the results in Table 4.56.

Algorithm	Recall	Precision	F1	|ER|	%ER
MBL	66.30	75.08	70.41	1.76	5.61
GIS-MaxEnt	64.62	75.57	69.66	1.88	5.83
L-BFGS-MaxEnt	71.58	71.98	71.78	1.43	4.82
SLIPPER	76.32	41.08	53.41	1.21	2.53

Table 4.56: Performance with Auxiliary-final question feature on combined data parsed with Charniak's parser

Traces 5 false negatives are due to traces being inserted instead of the ellipsed material marker.

Main verb As before, false positives are generated due to main verbs. 32 such cases occur. The increase in numbers is due to the degradation in parse structure, which results in features like Auxiliary-final VP classifying with less precision, and being overturned where there is a surface resemblance in the context.

Comparatives Comparatives give rise to 4 false positives.

Inversion 11 false positives occur due to inversion, an example of which is seen in Figure 4.29. These cases are easy to confuse with VPE given a limited context, but are resolved simply by a reordering of the sentence. The sentence in Figure 4.29, for example, can be resolved by moving the question to the beginning, giving "Isn't it sort of remorseless ?". Repetition of the antecedent, which would give "Sort of remorseless, isn't it sort of remorseless ?" is unnecessary.

Tagging/Parsing error There is one case of tagging error that gives rise to a false positive, seen in Figure 4.30. This type of tagging error, where a possessive is confused with an auxiliary, occurs often in the dataset, but does not give rise to errors in the VPE detection process in the rest of the cases.

```
(S1
  (SQ
    (VB Sort)
    (PP (IN of) (NP (NN remorseless)))
    (, ,)
    (SQ (VBZ is) (RB n't) (NP (PRP it))
      (. ?)) (. ?)))
```

Figure 4.29: Inversion

```
(S1
  (S
    (NP (PRP It))
    (VP (VBZ 's) (ADVP (RB just))
      (NP (NP (PRP$ my) (NN word))
        (PP (IN against)
          (S (NP (NN everybody) (RB else))
            (VP (VBZ 's)))))) (. !)))
```

Figure 4.30: Charniak mistag

The Empty VP feature is not reliable and has a tendency to be inserted before conjunctions (Figure 4.31).

There are also 3 cases where 'to' is parsed as being part of a VPE, where it is not (Figure 4.32).

Remaining cases The rest of the mistakes arise from some level of parse error. There are numerous cases where the VP header is used where SQ should, reducing its usefulness. For the rest of the 80 cases, one or more of the features usually do predict correctly, but are outweighed by the other features.

Cross-validation The results (Table 4.57) show a very small amount of improvement, with the top score still at 70% F1.

```
(S1
  (S
    (S (PP (IN At) (NP (DT any) (NN rate)))
      (NP (PRP you)) (VP (MD can) (RB not)
        (VP (VB stand) (ADVP (RB apart)))))
    (, ,)
    (S (NP (PRP I))
      (VP (VP (MD can) (RB not) (VP (-NONE- *?*)))
        (CC but) (VP (VBP do) (ADVP (RB otherwise)))))
    (CC and)
    (S (NP (DT that))
      (VP (VBZ is) (SBAR (WHADVP (WRB why))
        (S (NP (PRP we)) (VP (VBP belong)
          (ADVP (RB together)) (, ,)
          (SBAR (-NONE- 0)
          (S (NP (NNS d')) (VP (VBP ye) (VP (VB see))))))))) (. ?)))
```

Figure 4.31: Charniak Empty VP insertion

```
(S1
  (SBARQ ('' '')
    (WHNP (WP What))
    (SQ (VBP are) (NP (PRP you))
      (VP (IN up)
        (S (NP (-NONE- *)) (VP (TO to) (VP (-NONE- *?*))))) (. !)
('' '')))
```

Figure 4.32: Charniak wrong 'to' parse

Algorithm	Recall	Precision	F1	|ER|	%ER
MBL	66.25	69.38	67.78	0.46	1.41
GIS-MaxEnt	62.26	74.07	67.65	0.55	1.67
L-BFGS-MaxEnt	69.90	71.71	70.79	0.54	1.82
SLIPPER	74.29	40.33	52.28	2.47	4.92

Table 4.57: Cross-validation using combined dataset parsed with Charniak's parser, incorporating the Auxiliary-final question feature

```
(|T/frag|
 (|T/lmta_np| |Now:1_RT|
  (|Taph/comma+/-| |,:2_,|
   (|Tph/a2/-| (|AP/a1| (|A1/a| |apparently:3_RR|))) |,:4_,|))
 |they:5_PPHS2| |do:6_VD0|)
```

Figure 4.33: Incomplete parse from RASP - 1

```
(|T/txt-sc1/---|
 (|S/np_vp| |He:1_PPHS1|
  (|VP/vp_adv|
   (|VP/vp_pp| (V/0 |go+ed:2_VVD|)
    (|PP/p1|
     (|P1/p_np| |for:3_IF|
      (|T/lmta_np| (|NP/det_n| |the:4_AT| (|N1/n| |bed:5_NN1|))
       (|Taph/comma+/-| |,:6_,|
        (|Tph/vp/-|
         (|VP/cj_int/+|
          (|V/np| (|V/0_p| |jump+ed:7_VVD| |on:8_II|) |it:9_PPH1|)
          |,:10_,|
          (|VP/cj_end| |and:11_CC|
           (|V/s| |strike+ed:12_VVD|
            (|S/whpp-aux| (|PP/p1| (|P1/pwh| |where:13_RRQ|))
             |he:14_PPHS1| |could:15_VM|)))))
        |,:16_,|)))))
   (|AP/a1| (|A1/a| |repeatedly:17_RR|)))))
```

Figure 4.34: Incomplete parse from RASP - 2

4.5.3 RASP data

VP-final auxiliary in fragmented sentences

In 7 examples, such as the one seen in Figure 4.33, verb phrases were not marked out as such by RASP when a full parse is not generated, or, in 5 cases were only taken as part of a sentence without a VP header, as in Figure 4.34. This results in the Auxiliary-final VP feature not detecting these instances.

To alleviate this problem, we experimented with expanding the search space for clauses ending with auxiliaries to fragments of sentences (denoted by T/* in RASP) and further to sentences (denoted by S/* in RASP).

The effect of these modifications on the success rate of the feature itself (Table 4.58) is a 10% increase in F1 when checking for fragments as well as VPs. Increasing the search to sentences increases recall but at a higher cost to precision, making it overall less successful.

Feature	Recall	Precision	F1
VP	18.56	27.12	22.04
VP + Fragment	32.96	32.42	32.69
VP + Fragment + Sentence	36.29	26.46	30.61

Table 4.58: Correcting the auxiliary-final VP feature

The effect of the changes to classification accuracy when incorporated as a feature is not uniform (Table 4.59). The effect on SLIPPER varies wildly, while for the other three classifiers extending the search to fragments only provides improvements to all, and further extension to sentences reduces scores for two of the classifiers. Based on these results, only the extension to fragments will be used.

Algorithm	Aux in	Recall	Precision	F1	|ER|	%ER
MBL	V+F	63.71	71.20	67.25	0.10	0.30
	V+F+S	62.04	70.44	65.97	-1.18	-3.59
GIS-MaxEnt	V+F	61.77	75.85	68.09	0.71	2.18
	V+F+S	62.88	76.17	68.89	1.51	4.63
L-BFGS-MaxEnt	V+F	68.42	76.47	72.22	1.3	4.47
	V+F+S	67.31	75.23	71.05	0.13	0.45
SLIPPER	V+F	80.33	32.62	46.40	-11.06	-26.00
	V+F+S	74.79	49.09	59.28	1.82	4.28

Table 4.59: Results using corrected auxiliary-final VP feature

No parses In eight cases, RASP returns no parse. One of these has a parse for half the sentence, but not the second half, where the VPE occurs, making it difficult to detect such cases. Examples in sentences detected as not having a parse structure will have their auxiliary-final VP feature set to unknown. In SLIPPER and MBL this is achieved by returning '?', while in the Maximum Entropy classifiers not supplying a value for the feature suffices.

Auxiliary-final VP without a VP In 14 cases the pattern for an auxiliary-final VP feature is found connected to the top level of the sentence without a

```
(|T/frag|
 (|NP/cj_end/--| |But:1_CCB|
  (|NP/det_n| |my:2_APP$| (|N1/n| |daughter:3_NN1|)))
 |could:4_VM| |not+:5_XX| |,:6_,|
 (|Tph/pp/-|
  (|PP/p1|
   (|P1/p_s| |until:7_ICS|
    (|S/np_vp| |she:8_PPHS1|
     (|V/be_ap/-| |be+ed:9_VBDZ|
      (|AP/a1|
       (|A1/adv_a1| (|AP/a1| (|A1/a| |too:10_RG|))
        (|A1/a_inf| |ill:11_JJ|
         (|V/to_bse| |to:12_TO|
          (|V/np| |take:13_VV0| |it:14_PPH1|))))))))))
```

Figure 4.35: Context for auxiliary-final VP feature, without a VP

VP header, as seen in Figure 4.35. This is like the cases discussed previously, in that it occurs in fragmented sentences, but differs in that it is not at the end of a T/S header, but in the middle. A feature was developed to check for such cases, giving the results seen in Table 4.60. For sentences with no parse, this feature is also set to unknown. For MBL, as seen before, the addition of a low precision feature lowers performance, while the Maximum Entropy methods incorporate it better. SLIPPER ignores it.

Algorithm	Recall	Precision	F1	|ER|	%ER
Root aux-final VP	47.29	11.75	18.83		
MBL	63.43	71.12	67.06	-0.19	-0.58
GIS-MaxEnt	64.27	75.82	69.57	1.48	4.64
L-BFGS-MaxEnt	68.98	76.62	72.59	0.37	1.33
SLIPPER	80.33	32.62	46.40	0	0

Table 4.60: Effects of using the root-connected auxiliary-final VP feature

Main verb There are 28 cases where a main verb is confused with an auxiliary, resulting in false positives. This includes cases such as "That will do".

Comparatives Three comparatives are confused with VPE, resulting in false positives.

Inversion Five cases of inversion are confused with VPE, resulting in false positives.

Tagging errors In one example the verb 'can' is tagged as a non-modal verb by RASP. Further examination of the test-set shows this occurs 22 times, where one is correct, two should have been tagged as nouns, and the rest as modal verbs. The features rely on RASP having tagged auxiliaries correctly to perform their checks, and those that are not tagged so are not considered at all. To counter this, a lexical check for the word 'can' was added, accepting it as an auxiliary under all circumstances. Forcing these to be accepted as auxiliaries gives no changes to the results, meaning that these generalizations were learned by the algorithms anyway, and the changes will be discarded.

Remaining cases Among the 54 remaining cases, four errors are noticeable as being due to limited context such as "We can not, however, join the political chorus..". In this sentence, the main verb is too far away from the auxiliary.

Cross-validation Results in Table 4.61 show that the changes made produce no significant, or even uniform, improvement, and the top score is still around 70%.

Algorithm	Recall	Precision	F1	|ER|	%ER
MBL	64.58	71.11	67.69	0.39	1.19
GIS-MaxEnt	64.45	69.30	66.78	0.38	1.13
L-BFGS-MaxEnt	66.15	71.71	69.72	-0.32	-1.07
SLIPPER	81.44	28.93	42.69	-2.29	-4.16

Table 4.61: Cross-validation using combined dataset parsed with RASP, incorporating the expanded Auxiliary-final VP and Root Auxiliary-final VP features

4.6 Summary of Chapter

This chapter 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. To summarize the results :

- Experiments utilising six different machine learning algorithms have been conducted on the task of detecting VPE instances. Of these, two were abandoned due to technical problems with the implementation of the software (TBL) or because the sparse nature of the data was not suitable (decision trees), leaving MBL, SLIPPER, GIS-based and L-BFGS-based maximum entropy modelling for the final results.

- Two parsers were used, Charniak's and RASP. Charniak's parser is trained on the Treebank and uses a similar tagset. RASP is trained on a diverse set of data, and uses a tagset similar to the BNC.

- Using the lexical forms and Part of Speech (POS) tags of the words in the BNC 76% F1 is obtained.

- Experiments on the Penn Treebank show poorer performance for just lexical form and POS data (62%), due to the coarser tagset employed, but adding features derived from the extended syntactic information available improves results to 82% F1. The most informative feature extracted from the Treebank, Empty VPs, has a 72% F1. These results are a clear improvement over previous results, which achieved around 48% F1.

- Re-parsing this dataset to investigate performance using non-perfect data gives 67% F1 for both parsers. Charniak's parser combined with Johnson's algorithm generates the Empty VP feature with 32% F1. RASP, which does not have the Empty-VP feature, generates some of the other features more reliably.

- Repeating the experiments by parsing parts of the BNC gives 71% F1, with the empty VP feature further reduced to 25% F1, an expected drop due to Charniak's parser being trained on the Treebank only. Combining the datasets, final results of 71% F1 are obtained. These results are lower than results using the original BNC dataset with just the words and POS tags (76%). This is due to errors introduced in the parsing process; the results using the RASP-parsed BNC, with just the words and POS tags is 66%. Given that RASP uses the same tagset as the BNC, it can be seen that errors in the POS tags alone result in 10% lower F1. The syntactic features extracted are also error-prone, and do not improve the results greatly.

- The effect of the syntactic features used are, not surprisingly, correlated with their reliability. On the hand-annotated Treebank, they give an average of 59% %ER, with a 54% minimum. On all experiments with re-parsed data, the average %ER is 26%, but the minimum varies : 13% for the re-parsed Treebank, 8% on the parsed BNC, and 2% on the combined data.

- L-BFGS-MaxEnt is consistently the best performing classifier, with the results seen below. This is partly due to the strength of the Maximum Entropy framework, and partly due to the Gaussian Prior smoothing. Also, the use of L-BFGS parameter estimation gives improvements over GIS parameter estimation.

Corpus	Words + POS	+ features
BNC	76.01	-
Treebank	62.36	82.63
Charniak-Treebank	62.11	67.58
RASP-Treebank	62.02	67.24
Charniak-BNC	69.10	71.75
RASP-BNC	66.20	69.28
Charniak-Combined	68.06	70.25
RASP-Combined	67.25	70.04

- SLIPPER is consistently the worst performing classifier. This is due to the fact that the greedy learning algorithm employed in SLIPPER is not aimed at the kind of sparse datasets used in these experiments.

Corpus	Words + POS	+ features
Treebank	23.02	64.63
Charniak-Treebank	21.63	44.02
RASP-Treebank	17.38	43.23
Charniak-BNC	30.64	55.59
RASP-BNC	15.16	51.32
Charniak-Combined	23.95	49.81
RASP-Combined	14.47	44.98

- MBL and GIS-MaxEnt give similar results, with a few differences. MBL performs better with just words and POS tags, and performs better with RASP data.

	MBL		GIS-MaxEnt	
Corpus	Words + POS	+ features	Words + POS	+ features
BNC	72.04	-	67.58	-
Treebank	46.92	78.66	33.64	79.10
Charniak-Treebank	46.20	61.22	36.44	56.85
RASP-Treebank	57.60	66.24	33.10	63.09
Charniak-BNC	54.93	67.41	50.56	66.36
RASP-BNC	60.01	66.07	48.76	67.74
Charniak-Combined	53.87	67.32	51.10	67.10
RASP-Combined	61.19	67.30	47.45	66.40

- Charniak's parser generally outperforms RASP, but with too small a margin for a definitive conclusion. An interesting point is that while experiments on annotated data show that RASP's tagging scheme should produce superior results to Charniak's tagging scheme, experiments with the tagged data do not bear this out. This suggests that RASP introduces more error into the tagging process than Charniak's parser does.

- It was seen that the machine learning algorithms used do not take full advantage of features with high precision, and that running these tests separately produces better results. This, combined with error correction for systematic problems, improves the top score for the Treebank to 85%.

The results demonstrate that the method can be applied to practical tasks using free text, and that it is possible to achieve results using automatically parsed data that are not too degraded compared to hand-annotated data. This work offers clear improvement over previous work, and is the first to handle un-annotated free text, where VPE detection can still be done with good recall and precision.

As machine learning is used to combine various features, this method can be extended to other forms of ellipsis, and to ellipsis involving verbs in other languages. However, a number of the features used are specific to English VPE, and would have to be adapted to such cases. It is difficult to extrapolate how successful such approaches would be based on current work, but it can be expected that they would be feasible, albeit with lower performance.

Further work can be done on extracting grammatical relation information from the Treebank (Lappin et al., 1989; Cahill et al., 2002), or using those provided by RASP, to generate 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. Also, since the start of the work new methods for inserting empty category information have been developed, and utilizing these may improve results.

Looking at the patterns for VPE occurrence in the data, it is also observed that there is a weak clustering of these occurrences. The frequency with which ellipsis is used is particular to each speaker/writer, and past occurrences of ellipsis can indicate increased likelihood of further ellipsis in a section or by a speaker. This could be used in a feature in the form of a decaying exponential that increases the likelihood of VPE. If the data includes dialogue, a dialogue tracking system may give improvements to this feature by increasing the VPE utterance likelihood of individual speakers. A further refinement to this could be to condition the probabilities on the syntactic and semantic contexts for VPE preferred by each speaker.

Two paths of work were intentionally left unexplored, as any gains they may have provided would not have been large enough to alter the conclusions I have arrived at. The μ-TBL package gave promising results but had to be abandoned for technical reasons. Time could have been spent on fixing this, or writing a new package, but this would have been beyond the scope of this work. The reason for using several ML packages is to validate the usefulness of the data, shown for example by correlated increases in performance when a feature is added. As stated before, the purpose of this work is not to compare ML methodologies.

It would also have been possible to combine the strengths of the two parsers used, by combining the features generated by both parsers, and only using the most successful one. This would have meant using the POS tags, and features reliant on POS tags, from RASP, and the rest from Charniak's parser. This was not done because it would simply have been an attempt to counter deficiencies in the parsers, and would have provided little insight that is not already obtained from comparing results with the hand-annotated data, i.e. better parsers will produce better results. Furthermore, it can even be suggested that the performance on

parsed data could have been roughly guessed by multiplying the parsing accuracy with the performance on hand-annotated data, which does fit with the results presented in this work. Given that parsing is an area of intensive research, it can only be expected that as parsers get better, results obtained through the methods used in this work, which are parser-independent, will gravitate towards the upper bound set by hand-annotated data.

Chapter 5

Identifying the antecedent

In this chapter, work done on the second stage of the VPE resolution system is described. Section 5.1 describes previous work on the topic.

Section 5.2 describes the benchmark algorithm. This is given a separate section as it is the result of previous work and is described in detail.

Section 5.3 describes experiments done using the Penn Treebank. These experiments form the bulk of the work, and describe attempts at improving the benchmark algorithm through the use of ML techniques and added features.

In section 5.4, the algorithms of the previous section are applied to automatically parsed data.

5.1 Previous work

Hardt's (1992a; 1997) syntactic algorithm is, to our knowledge, the only algorithm to have been empirically tested for the task of antecedent extraction. The algorithm uses constraints and preference factors to determine possible antecedents of VPE occurrences from the Penn Treebank. The paper claims a success rate of 94.8%, compared to a 75.0% recency-based baseline, where success is defined as sharing of a head between the system result and human choice. The results for a success criterion of word-for-word match with human choice is 76.0%, compared to a 14.6% baseline.

Approaches similar to those that will be pursued in this thesis have been applied to anaphora resolution, such as Lappin and Leass (1994).

Rambow and Hardt (2001) present work on VPE generation. They use a variety of surface, morphological, semantic and discourse features for ML. Some of these features will be adapted to the antecedent detection experiments in section 5.3.

5.2 Benchmark algorithm

I will use Hardt's (1997) VPE-RES algorithm for the Penn Treebank as a benchmark. A description of this algorithm is seen below, including details not found in the published papers. The original LISP code was kindly made available to me by Daniel Hardt, which facilitated greatly the process.

1. **Populate antecedents** All Verb Phrases (VPs) in the sentence the VPE is in, including those following it, and all VPs in the two previous sentences, are taken as possible antecedents. Each antecedent has a score associated with it, initialised to 1.

2. **Remove unresolved VPEs** Antecedents that are other VPE sites which were not previously resolved are removed from the list of possible antecedents.

3. **Remove auxiliary VPs** Antecedents beginning with auxiliaries are removed. There are special checks for antecedents beginning with 'ought' or 'let' and lacking a subject, which are removed as well.

4. **Syntactic filter** Antecedents containing the VPE in a sentential complement are ruled out. The clause containing the VPE is not considered to be a sentential complement if it begins with :

    - what/who/which (WHNP).
    - an adverbial phrase (ADV).
    - a subject-auxiliary inversion (SINV).
    - when/where (a restricted version of WHADVP)

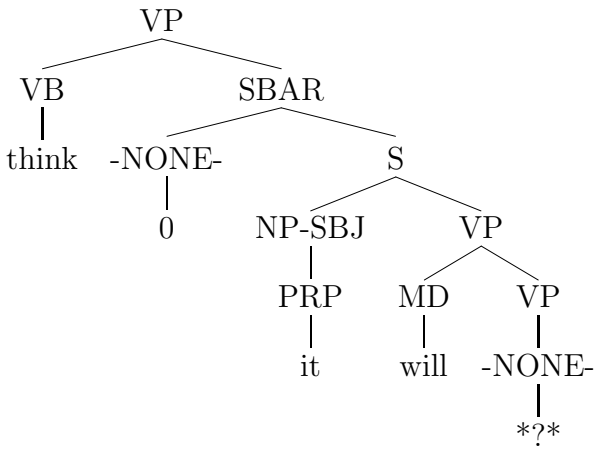

Figure 5.1: Antecedent ruled out by syntactic filter

- a quantifier phrase (QP)
- an adverb (RB).
- a preposition or subordinating conjunction (IN), except 'that'.

This filter will block incorrect antecedents such as the one in Figure 5.1 ('think'). Extra checks are made to rule out the following related instances as well :

- Using the definition of sentential complement outlined above, antecedents consisting of an adjectival phrase containing the VPE in a sentential complement are ruled out. This is intended to rule out instances such as "am sure that they do".
- Antecedents starting with 'as' and followed by the VPE in a sentential complement are ruled out.

5. Clausal recency Based on their distance from the VPE site, antecedents have their score adjusted in the following manner : the score for the antecedent the farthest before the VPE is multiplied by 1.15, and each following that by powers of 1.15, giving 1.15, 1.32, 1.52 etc., so that the closer an antecedent is to the VPE site the higher its score. The first antecedent after the VPE gets a weight of 0.575, and each one after that is multiplied by powers of 0.667. If an antecedent is contained within another antecedent (ie. is a phrase constituent of), they are given the same recency score.

6. **SBAR relation** Antecedents containing a relative or subordinate clause (SBAR) that contains the VPE get their score boosted by a factor of 10, generally assuring that they are selected. The SBAR has to be directly below the VP head of the antecedent, or at most one level further below. This rule operates in conjunction with the syntactic filter, on antecedents not removed by the filter despite containing the VPE. An example of this preference factor in use is seen in Figure 5.2. Without the SBAR preference, the VP headed by 'feeling' would be chosen due to recency.

7. **Comparative relation** This preference factor is active when the antecedent contains the VPE in a comparative relation. This is signalled by a 'than' preceding the VPE, with no intervening VPs or the word 'as' between the antecedent head and the VPE. When this relation is found, the score of the antecedent is multiplied by 10.

8. **In quotes** If the VPE is within quotes, any antecedents that are not also in quotes have a penalty of 0.667 applied.

9. **Be-do match** If the VPE is a variant of 'do', antecedents with an auxiliary of type 'be' have a penalty of 0.667 applied.

10. **Auxiliary match** This rule applies to the auxiliaries of the larger VPs which contain the antecedents. Dividing auxiliaries into the base groups of do, be, have, can, would, should and to, antecedents which match the VPE in the form of their auxiliaries are given a boost of 1.5.

11. **Pick top choice** The antecedent with the highest score at this stage is chosen as the system's result. In the case of a tied score, the most recent antecedent is chosen.

12. **Post processing** Several steps of post processing are done, greatly improving the Exact Match score.

    - If the antecedent contains the VPE, the argument or adjunct containing the VPE is removed, as well as anything following this. This also serves to avoid situations of Antecedent Contained Ellipsis (ACE, sometimes also called Antecedent Contained Deletion). An example of such a case is seen in Figure 5.3. Here, the antecedent VP is *acted*

```
...
(NP
 (NP (DT an) (JJ exciting)
   (, ,)
   (JJ eclectic) (NN score) )
 (SBAR
   (WHNP-67 (WDT that) )
   (S
     (NP-SBJ (-NONE- *T*-67) )
     ( VP  (VBZ tells)
       (NP (PRP you) )
       (SBAR-NOM
         (WHNP-2 (WP what) )
         (S
           (NP-SBJ (DT the) (NNS characters) )
           (VP (VBP are)
             (VP
               (VP (VBG thinking)
                 (NP (-NONE- *T*-2) ))
               (CC and)
               (VP (VBG feeling)
                 (NP (-NONE- *T*-2) ))))))
       (ADVP-MNR
         (ADVP (RB far) (RBR more) (RB precisely) )
         ( SBAR  (IN than)
           (S
             (NP-SBJ
               (NP (NNS intertitles) )
               (, ,) (CC or)
               (NP (RB even) (NNS words) )
               (, ,) )
             (VP (MD would)
               (VP (-NONE- *?*) ))))))))))
   (. .) ))
```

Figure 5.2: Antecedent given priority by SBAR-relation factor

```
    ...
(NP-SBJ (PRP I) )
(VP
  (VP (VBD was)
    (ADJP-PRD (JJ shy) )
    (PP (IN with)
      (NP (NNP Jessie) )))
  (CC and)
  (VP (VBD acted)
    (SBAR-ADV (IN as)
      (S
        (NP-SBJ (PRP I) )
        (VP (VBD had)
          (VP (-NONE- *?*)
            (PP-TMP (IN during)
              (NP
                (NP (DT those) (JJ early)
                   (NNP Saturday) (NNS mornings) )
                      ...
```

Figure 5.3: Antecedent contained ellipsis

as I had during those early Saturday mornings, but after the filter only *acted* remains, which gives the correct resolution.

- If the following phrase is adverbial, it is added to the tail of the antecedent.
- The word 'not' is removed from the beginning.
- The words 'anymore', 'yet', 'too', 'as' and 'even' are removed from the end.

The advantage of this scoring system is that most of the preference factors encode just a preference and not hard rules, resulting in flexibility[1].

For the present work, a number of differences have been implemented :

[1]For example, it may seem logical at first glance to generate hard rules for auxiliary form matching as well as preferences. A sentence from the corpus is seen below :

(37) It's awkward, you see that, isn't it ?

The assumption here would be that the antecedent is *awkward*. Looking at past utterances by the same speaker, however, shows that he uses *isn't it* with little regard for auxiliary matching:

(38) a. "Sure, sure, you're the one take over for Pretty, soon as I get the supply, get started up again, isn't it ?"

- Due to the different versions of the Treebank used, minor changes have been made to account for changes in category names. Antecedents are not just VPs, but can also be SQs (as discussed in the previous chapter).

- The two previous sentence limitation was used by VPE-RES, as there were no observed instances of antecedents further away. Both the version of the Treebank and the BNC in the current experiments, however, contain antecedents outside this window. There are antecedents that follow the VPE, by up to four sentences. Antecedents occurring after the VPE generally tend to be caused by cut-off sentences in the data, as in :

  (39) "I don't - " he began, cleared his throat, "I don't usually dance all that much".

  There are also 40 antecedents that are between three and six sentences back. Furthermore, there are rare instances of VPEs with antecedents as far back as up to 15 sentences. These last ones are, to an extent, extreme cases, and are difficult to resolve even by a reader, without going back to search for the antecedent. Most cases with a large number of sentences between the VPE site and antecedent come from parts of the data with dialogue between characters. The algorithm was modified to accept an argument to look as far back and forward as desired.

- The VPE-RES algorithm removes any unresolved VPEs from the possible antecedents list, and is coded to resolve to the previous VPE in the chain. In practise, it appears that the corpora used in the original VPE-RES experiments were modified to deal with multiple VPEs sharing an antecedent. The sentences containing these are presented twice in the data, where the first copy only has the first VPE marked (Figure 5.4) , and the second copy only has the second VPE marked (Figure 5.5). It should be noted that for these datasets VPEs were marked by the VPE marker heading VPs.

b. "Two weeks, a month, we talk it over again, and maybe if nothing happens meanwhile to say the cops know this and that, then we make a little deal, isn't it ?"

It can now be inferred that the correct antecedent for (37) is *see that*, and *isn't it* is meant to stand for *don't you*.

```
(TOP (S (NP-SBJ-2 (PRP One))
        (VP (VP (VBZ learns)
                (NP (NP (DT a)
                        (NN lot))
                    (PP (-NONE- *ICH*-1)))
                (PP-CLR (IN from)
                        (NP (DT this)
                            (NN book))))
            (  )
            (CC or)
            (VP (VBZ seems)
                (S (NP-SBJ (-NONE- *-2))
                   (VP (TO to)
                       (VPE (-NONE- *?*)))))
            (  )
            (PP-1 (IN about)
                  (NP (JJ crippling)
                      (JJ federal)
                      (NN bureaucracy))))
        (   )))
(TOP (FRAG (     _oquote )
           (FRAG (S (NP-SBJ-1 (-NONE- *))
                    (VP (VBZ Seems)
                        (S (NP-SBJ (-NONE- *-1))
                           (VP (TO to)
                               (VP (-NONE- *?*)))))))
           ( _cquote )
           (SBAR-PRP (IN because)
                ...
```

Figure 5.4: Chained anaphora - First Instance

I opt to resolve to the original antecedent site. This is not always correct, as (40) shows. The correct way of resolving such chained (inherited) cases would be to use the resolved versions of the VPs as antecedents, but the problem with this approach is that errors may propagate. As there are very few instances of chained VPEs in the corpora (three in all), and removing all previous VPEs results in an overall improvement in performance, I remove all VPEs.

(40) I must confess that in all the times I read Madame Bovary, I never noticed the heroine's rainbow eyes.

```
(TOP (S (NP-SBJ-2 (PRP One))
        (VP (VP (VBZ learns)
                (NP (NP (DT a)
                        (NN lot))
                    (PP (-NONE- *ICH*-1)))
                (PP-CLR (IN from)
                        (NP (DT this)
                            (NN book))))
            (  )
            (CC or)
            (VP (VBZ seems)
                (S (NP-SBJ (-NONE- *-2))
                   (VP (TO to)
                       (VP) (-NONE- *?*)))))
            (  )
            (PP-1 (IN about)
                  (NP (JJ crippling)
                      (JJ federal)
                      (NN bureaucracy))))
        (   )))
(TOP (FRAG (     _oquote )
           (FRAG (S (NP-SBJ-1 (-NONE- *))
                    (VP (VBZ Seems)
                        (S (NP-SBJ (-NONE- *-1))
                           (VP (TO to)
                               (VPE) (-NONE- *?*)))))))
           ( _cquote )
           (SBAR-PRP (IN because)
                ...
```

Figure 5.5: Chained anaphora - Second Instance

Should I have ? [noticed the heroine's rainbow eyes]
Would you ? [have noticed the heroine's rainbow eyes]

- The check for words to remove from the end of antecedents has been expanded to include punctuation, and 'of course'.

- In the Brown section of the original corpus, empty sentences are placed between text units to limit the search. This was not seen to affect performance on initial experiments with the current corpora, and was not used.

- An automatic scoring function for the success criteria has been implemented. The method of searching for the Head Verb for scoring is taken from Daniel Hardt's code.

5.3 Experiments using the Treebank data

5.3.1 Benchmark performance

The results for the original VPE-RES implementation on the intersection data (see section 3.2) are given in Table 5.1[2]. The performance on the WSJ and Brown data is similar, and in line with Hardt's findings.

	WSJ			Brown			Combined		
Criterion	#	%	Σ %	#	%	Σ %	#	%	Σ %
Exact Match	14	66.67	66.67	29	78.38	78.38	43	74.14	74.14
Head Match	4	19.05	85.71	4	10.81	89.19	8	13.79	87.93
Head Overlap	0	0	85.71	0	0	89.19	0	0	87.93
Miss	3	14.29	14.29	4	10.81	10.81	7	12.07	12.07

Table 5.1: Original-VPE-RES performance on intersection data

Using my re-implementation of the VPE-RES algorithm (New-VPE-RES) on the current version of the Treebank, the results in Table 5.2 are obtained. It is seen that performance is lower according to Exact Match, identical according to Head Match, and higher according to Head Overlap. These differences are due to changes made to the Treebank annotation scheme, and show that the new implementation works as expected.

The results on the whole test corpus (Table 5.3) show scores lower by about 10% compared to those for the intersection corpus. While the results for Head Overlap are comparable, Exact Match is significantly lower. These results are obtained using the original range of 2 sentences back. Increasing the range to 15 sentences back and 5 ahead gives a small improvement (Table 5.4).

[2](#) corresponds to the number of samples identified uniquely by each criteria. (%) is the corresponding percentage of the data. (Σ %) is the cumulative percentage; Head Match includes Exact Match, as its definition subsumes it, and Head Overlap contains both of them.

Criterion	#	%	Σ %
Exact Match	40	68.97	68.97
Head Match	11	18.97	87.93
Head Overlap	4	6.90	94.83
Miss	3	5.17	5.17

Table 5.2: New-VPE-RES performance on intersection corpus

Analysing the errors, it is seen that ten of them are due to antecedents being too far back. Nine are due to inconsistencies in the parse structures which result in the syntactic filter, auxiliary match and SBAR-match not working correctly. The final mistake is due to a missing VP header for the correct antecedent. The mistakes due to recency are due to the built-in bias to favour more recent antecedents, and may suggest a certain amount of overfitting on the data, or the lack of appropriate features to handle these cases.

Criterion	#	%	Σ %
Exact Match	92	61.33	61.33
Head Match	27	18.00	79.33
Head Overlap	9	6.00	85.33
Miss	22	14.67	14.67

Table 5.3: New-VPE-RES performance on test corpus

Criterion	#	%	Σ %
Exact Match	94	62.67	62.67
Head Match	27	18.00	80.67
Head Overlap	9	6.00	86.67
Miss	20	13.33	13.33

Table 5.4: New-VPE-RES performance on test corpus - increased range

Analysing the errors on the test data results, it is seen that Exact Match and Head Match may not be reliable indicators of performance. Most of the examples that are successful according to Head Overlap but not the two others are due to coder preference and do not indicate absolute judgements. One example is seen below.

(41) a. You know I don't want to see Pedersen.

b. Why should I ?

Here, the coder choice is "want to see Pedersen", and the system choice is "see Pedersen". Either choice is plausible without further information. Due to the apparent brittleness of Exact Match and Head Match, and the variability of speakers' judgements concerning VPE antecedent identification, I will use Head Overlap as my main success criterion in the following experiments.

5.3.1.1 Clausal recency weight

As half the errors present are due to the correct antecedents not being selected because they are too far away from the antecedent site, I experimented with various recency preference factor settings. Table 5.5 shows that the original setting of 1.15 produces the best results.

Criterion	1.05	1.10	1.15-1.20	1.25-1.30	1.35
Exact Match	58.00	59.33	**62.67**	61.33	60.67
Head Match	74.67	76.67	**80.67**	79.33	78.67
Head Overlap	80.00	82.67	**86.67**	86.00	85.33
Miss	20.00	17.33	13.33	14.00	14.67

Table 5.5: Effect of recency preference factor (Σ %)

5.3.1.2 Nested antecedents

Antecedents which are contained within another antecedent are given the same recency score by the algorithm. It is possible, however, to encounter an example such as (41), where "want to see Pedersen" and "see Pedersen" will get the same recency score, and none of the other features indicate which one is correct. Changing the recency procedure to ignore nesting of antecedents and apply the preference factor regardless will result in "see Pedersen" getting a higher weight than"want to see Pedersen", as it is closer to the VPE site. Results of using this approach are seen in Table 5.6. This approach diminishes results for the benchmark algorithm and will not be retained.

Criterion	#	%	Σ %
Exact Match	92	61.33	61.33
Head Match	27	18.00	79.33
Head Overlap	10	6.67	86.00
Miss	21	14.00	14.00

Table 5.6: New-VPE-RES performance on test corpus without antecedent nesting

5.3.2 Using ML to choose from list of antecedents

All classifiers previously used except TBL will be used for the ML experiments[3]. Instead of the monolithic scoring system of VPE-RES, I will attempt to convert its components into features that can be used for machine learning. This is achieved with the feature vector :

(42) recency=n sbar-rel=yes/no comp-rel=yes/no aux-match=yes/no be-do-mismatch=yes/no not-in-quotes=yes/no TRUE/FALSE

Clausal recency is given as an integer, with 1 being right before the VPE, 2 is two before it and so on. Antecedents after the VPE have distances of -1, -2 etc. The preference factors SBAR-relation, Comparative-relation, Auxiliary-match, Be-do mismatch and Not-in-quotes are all binary values. The syntactic filter and post-processing of the antecedent forms are used as before.

Each possible antecedent for each VPE is used in the training data, with only the Head Overlap criteria deciding whether it is acceptable as an antecedent or not. This means that for each VPE, there may be more than one correct antecedent given in the training data.

After classification by the ML systems, the confidence estimates for all antecedents for each VPE in the test set are compared, with the highest scoring one being selected as the system choice. If scores are tied, the most recent antecedent is chosen. This is then graded according to the three success criteria.

[3]In the period between commencing the experiments on VPE detection and initiating experiments on this section, several of the classifiers have been updated. For the following experiments, TiMBL 5.1, GIS-MaxEnt 2.3.0 and L-BFGS-MaxEnt 20041229 are used.

5.3.3 ML baseline

Results using the benchmark features in the classifiers are seen in Table 5.7e. The results are consistently lower than the benchmark. The reason for this can be seen by looking at the performance of the individual features on the test dataset (Table 5.8).

It is seen that using the most recent two antecedents can give an F1 of 68%[4]. While this feature is clearly very useful, it presents a problem for machine learning, as its effective range is rather large, at 189 values. This means that while associations may be formed with antecedents at distances close to 1, it may be difficult to learn that an antecedent with a distance of 6 is still preferrable to one with a distance of 7, all else being equal.

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
Exact Match	48.67	**51.33**	50.67	**51.33**	48.67
Head Match	69.33	69.33	71.33	**72.00**	64.00
Head Overlap	**82.00**	76.67	78.67	79.33	71.33
Miss	18.00	23.33	21.33	20.67	28.67

Table 5.7: Machine Learning performance on benchmark equivalent (Σ %)

Feature	Recall	Precision	F1
Recency=1	51.63	79.29	62.54
Recency=1-2	79.07	59.65	68.00
Recency=1-3	92.09	45.73	61.11
SBAR-rel	21.86	77.05	34.06
Comp-rel	1.86	100.00	3.65
Aux-match	22.33	8.09	11.88
Be-do mismatch	2.79	1.03	1.50
Not-in-quotes	1.86	0.27	0.47

Table 5.8: Performance of benchmark features

[4]This score should not be considered as a separate baseline independent of the VPE-RES algorithm. While it does not use the other preference factors, the antecedents have been filtered using the syntactic filter and are not simply the most recent VPs. Also, this is not a straightforward F1 score, but rather F1 over the corpus where multiple antecedents can be 'correct', as they are judged by the Head Overlap measure. These multiple choices would have to be narrowed down to a single one to get the real score.

5.3.4 Nested antecedents

Repeating the experiment of not giving the same recency scores to nested antecedents, the results in Table 5.9 are obtained. These results are generally better than those before, and this form of recency weighting will be retained for the machine learning experiments.

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
Exact Match	54.00	**56.67**	52.00	54.67	50.67
Head Match	70.00	**72.67**	70.67	70.00	64.67
Head Overlap	80.67	**82.67**	80.67	80.00	74.67
Miss	19.33	17.33	19.33	20.00	25.33

Table 5.9: Machine Learning performance on benchmark equivalent without antecedent nesting (Σ %)

5.3.5 Grouping recency

When recency information is removed, results are seen to drop dramatically (Table 5.10). Given the importance of this surface feature, it would be desirable to use it as effectively as possible.

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
Exact Match	18.67	**26.00**	**26.00**	**26.00**	19.33
Head Match	26.67	**34.67**	**34.67**	**34.67**	27.33
Head Overlap	27.33	**35.33**	**35.33**	**35.33**	28.00
Miss	72.67	64.67	64.67	64.67	72.00

Table 5.10: Performance with no recency information (Σ %)

As having a full range for recency generates quite a large range of values for the classifiers, I experimented with trying to limit this information. The results in Table 5.11 are for four experiments :

1. The most recent antecedent forms one group, and all the rest another group
2. The two most recent antecedents form one group, and all the rest another group

3. The two most recent antecedent each form one group, and all the rest another group

4. In addition to the feature above (3), the normal recency feature is used

It is seen that recency information stating whether an antecedent is the closest to the VPE site or the second closest is used, but the rest is mostly ignored. Experiments with grouping ranges of recencies, in groups of 2 to 10 have also shown that no information beyond the closest two antecedents is used. These experiments do not yield any improvements, so recency will be used as before.

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
If *recency* = 1 then *grouped recency* = 1, else *grouped recency* = 2					
Exact Match	**50.00**	**50.00**	49.33	49.33	49.33
Head Match	64.00	**64.67**	64.00	64.00	63.33
Head Overlap	74.00	**74.67**	74.00	74.00	72.67
Miss	26.00	25.33	26.00	26.00	27.33
If *recency* < 3 then *grouped recency* = 1, else *grouped recency* = 2					
Exact Match	37.33	**43.33**	**43.33**	**43.33**	20.00
Head Match	54.67	**57.33**	**57.33**	**57.33**	28.00
Head Overlap	64.00	**61.33**	**61.33**	**61.33**	28.00
Miss	36.00	38.67	38.67	38.67	72.00
If *recency* < 3 then *grouped recency* = *recency*, else *grouped recency* = 3					
Exact Match	52.00	**56.67**	54.67	54.67	50.67
Head Match	66.67	**72.00**	72.67	70.00	64.67
Head Overlap	76.67	82.00	**82.67**	80.00	74.67
Miss	23.33	18.00	17.33	20.00	25.33
grouped recency as above, but with recency as a separate feature as well					
Exact Match	52.00	**56.67**	54.67	54.67	50.67
Head Match	66.67	**72.00**	72.67	70.00	64.67
Head Overlap	76.67	82.00	**82.67**	80.00	74.67
Miss	23.33	18.00	17.33	20.00	25.33

Table 5.11: Grouping the recency feature (Σ %)

5.3.6 Sentential distance

In order to see whether the clausal recency measure can be aided by a coarser grained distance metric, I conducted an experiment adding sentence based distance. This feature is an integer value of the number of sentences between the

antecedent and VPE sites; the value is zero if they are in the same sentence. It is seen to offer no improvements over the normal recency feature alone (Table 5.12).

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
Exact Match	53.33	**56.67**	52.00	54.00	51.33
Head Match	68.67	**72.67**	70.67	70.00	66.00
Head Overlap	78.67	**82.67**	80.67	80.00	75.33
Miss	21.33	17.33	19.33	20.00	24.67

Table 5.12: Sentential distance (Σ %)

5.3.7 Word distance

Adding a purely word based distance to the recency feature (Table 5.13) and the sentential distance feature as well (Table 5.14) is seen to offer a small amount of improvement.

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
Exact Match	**60.00**	52.67	51.33	56.00	56.00
Head Match	**76.00**	71.33	70.67	72.67	70.00
Head Overlap	**83.33**	82.00	81.33	81.33	76.00
Miss	16.67	18.00	18.67	18.67	24.00

Table 5.13: Word distance with recency (Σ %)

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
Exact Match	**60.00**	53.33	51.33	56.67	56.00
Head Match	**76.00**	71.33	70.67	73.33	71.33
Head Overlap	**83.33**	79.33	81.33	81.33	78.00
Miss	16.67	20.67	18.67	18.67	22.00

Table 5.14: Word distance with sentential distance and recency (Σ %)

5.3.8 Antecedent size

To investigate correlation between VP sizes and the likelihood of being the correct antecedent of a VPE, the number of words each antecedent consists of is added

as a feature. This feature does not give improvements (Table 5.15).

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
Exact Match	**58.67**	48.67	52.00	56.67	47.33
Head Match	**75.33**	68.00	70.67	74.00	58.67
Head Overlap	**82.67**	79.33	81.33	82.00	68.00
Miss	17.33	20.67	18.67	18.00	32.00

Table 5.15: Antecedent size (Σ %)

5.3.9 Auxiliary forms

Two features, *auxiliary match* and *be-do match*, already capture preferences based on the auxiliaries of the VPE site and the antecedent. The first experiment in Table 5.16 shows results without these features, which is seen to be slightly degraded. The second experiment provides the auxiliary forms of the antecedent and VPE as features, in one of eight categories[5]. The final experiment uses the auxiliary forms as well as the *auxiliary match* and *be-do match* features.

What can be seen from these experiments is that the *auxiliary match* and *be-do match* features do not have a high impact on the final results. The auxiliary forms themselves as features do not offer much improvement, and the *auxiliary match* and *be-do match* derived from them is still more informative. Using the auxiliary forms as well as the derived features offers no improvement over the derived features alone.

5.3.10 As-appositives

In order to investigate whether VPEs occurring in as-appositive constructions, as in (43), may be disposed towards a certain type of antecedent, a feature was added signifying whether this construction occurs by checking for the word 'as' in the few words prior to the VPE. This feature gives a slight improvement over previous results.

[5]Do, be, have, can, would, should, to, and not found

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
No auxiliary related information					
Exact Match	**53.33**	52.00	52.00	52.67	50.00
Head Match	66.67	69.33	**70.67**	69.33	67.33
Head Overlap	76.00	78.67	**81.33**	80.00	78.67
Miss	24.00	21.33	18.67	20.00	21.33
Auxiliary forms alone					
Exact Match	54.00	48.67	52.00	**54.67**	50.67
Head Match	68.67	66.67	70.67	**72.00**	70.00
Head Overlap	78.67	76.67	**81.33**	80.67	78.00
Miss	21.33	23.33	18.67	19.33	22.00
Auxiliary forms with *auxiliary match* and *be-do match*					
Exact Match	56.00	49.33	51.33	**56.67**	52.00
Head Match	72.67	66.00	70.67	**74.67**	66.00
Head Overlap	**82.00**	76.67	81.33	**82.00**	74.67
Miss	18.00	23.33	18.67	18.00	25.33

Table 5.16: Auxiliary form experiments (Σ %)

(43) It operated on, by and for the people individually just as did the Federal Constitution.

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
Exact Match	55.33	50.00	51.33	**56.67**	52.67
Head Match	71.33	66.67	70.67	**75.33**	71.33
Head Overlap	80.67	77.33	81.33	**82.67**	80.67
Miss	19.33	22.67	18.67	17.33	19.33

Table 5.17: As-appositives (Σ %)

5.3.11 Polarity

Table 5.18 shows results on experiments conducted to determine whether the polarity of the sentences containing the antecedent and VPE are informative. This check is done by searching for 'not' or one of its contractions in the respective sentences. It is seen that the antecedent site polarity alone does not affect results, while the VPE site polarity alone does. Using both is still better, but best results are obtained by using the boolean disjunction of the features. A more

complicated feature that combines all of these into four values does not give further improvement.

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
VPE site polarity alone					
Exact Match	**56.67**	52.67	51.33	56.00	54.67
Head Match	72.67	67.33	70.67	**75.33**	70.00
Head Overlap	82.67	78.67	81.33	**83.33**	75.33
Miss	17.33	21.33	18.67	16.67	24.67
Antecedent site polarity alone					
Exact Match	54.00	54.00	51.33	**56.67**	50.67
Head Match	70.67	72.00	70.67	**75.33**	70.67
Head Overlap	80.00	81.33	81.33	**82.67**	79.33
Miss	20.00	18.67	18.67	17.33	20.67
Both polarities					
Exact Match	56.00	54.00	51.33	**56.67**	56.67
Head Match	72.00	71.33	70.67	**76.00**	72.67
Head Overlap	81.33	81.33	81.33	**83.33**	78.00
Miss	18.67	18.67	18.67	16.67	22.00
VPE site polarity OR Antecedent site polarity					
Exact Match	54.67	54.00	51.33	**57.33**	51.33
Head Match	71.33	69.33	70.67	**76.67**	65.33
Head Overlap	80.67	79.33	81.33	**83.33**	74.00
Miss	19.33	20.67	18.67	16.67	26.00
Four valued : Both, VPE alone, Antecedent alone, None					
Exact Match	51.33	54.67	51.33	**56.67**	52.00
Head Match	68.67	72.00	70.67	**75.33**	69.33
Head Overlap	79.33	81.33	81.33	**83.33**	76.00
Miss	20.67	18.67	18.67	16.67	24.00

Table 5.18: Polarity (Σ %)

5.3.12 Adjuncts

This feature checks for adjuncts to the VPs containing the antecedent and VPE. The values for it are :

- Neither has an adjunct
- VPE alone has an adjunct
- Antecedent alone has an adjunct

- Both VPE and antecedent have adjuncts, and they are identical
- Both VPE and antecedent have adjuncts, and they are not identical

This feature gives a slight improvement (Table 5.19). The check for whether the adjuncts are identical is a simple string comparison. It would be desirable to have a more flexible measure of identity. Including semantic class information, from a source such as WordNet (Miller, 1995), might also be useful. These ideas will be left for future work.

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
Exact Match	53.33	56.00	52.00	**57.33**	54.00
Head Match	68.00	72.00	71.33	**76.00**	72.00
Head Overlap	78.00	80.67	81.33	**84.00**	79.33
Miss	22.00	19.33	18.67	16.00	20.67

Table 5.19: Adjuncts (Σ %)

5.3.13 Coordination

This feature checks whether the antecedent and VPE are in phrases coordinated by conjunction. The two italicized phrases in (44) are coordinated, and using this feature, it may be possible to form a preference which would choose "used by their husbands" over the more recent "indicates". There is no limit on the depth the VPs containing the antecedent and VPE can be in the phrases being coordinated. This feature only looks for intra-sentence coordination, and inter-sentence coordination is not dealt with.

(44) *Wives of the period shamefacedly thought of themselves as "used" by their husbands* – and, *history indicates, they often quite literally were.* ["used" by their husbands]

Unfortunately, there is only one example of coordination in the test corpus of the held-out test, which is not enough to give a general idea of the usefulness of the feature. Adding the feature results in no improvement (Table 5.20).

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
Exact Match	54.00	56.00	52.00	**57.33**	56.67
Head Match	69.33	72.00	70.67	**75.33**	76.00
Head Overlap	78.67	80.67	80.67	**82.67**	80.00
Miss	21.33	19.33	19.33	17.33	20.00

Table 5.20: Coordination (Σ %)

5.3.14 Subject matching

Examination of the mistakes encountered in the test corpus shows a feature that would be beneficial: subject matching. Examples where it would be useful are seen in (45).

(45) a. He tried to stifle it. But the words were forming. He knew he couldn't. [stifle it]

b. You want the kid to die ? Do you ? [want the kid to die]

c. " Do you want to call Eugene ?"
He didn't, but it was not really a question, and so he left the room, walked down the hall to the front of the apartment, hesitated, and then knocked lightly on the closed door of the study. [want to call Eugene]

Here, because of recency, 'forming', 'die' and 'call Eugene' will be chosen as the antecedents, despite the parallelism between the VPE sites and the correct antecedents suggesting otherwise. There are 6 such instances in the test data. For this task to be done reliably, a pronoun resolution system is necessary to match the subjects. I investigated using JavaRAP (Qiu et al., 2004) and MARS (Mitkov et al., 2002). Other available anaphora resolution implementations I reviewed required proprietary parsers or chunkers, or weren't available online for demonstration purposes.

Both systems only handle 3rd person pronouns, which covers only 2 of the cases. Of these, each gets one right and one wrong. As input data, the sentence the anaphor occurs in, plus the 5 sentences before and after it were used. It is not possible to form a conclusive judgement without annotating the whole corpus

using a feature derived with an anaphora resolution tool, or without trying other tools, but this initial examination suggests that, for the purposes of VPE antecedent location, automated methods may not be straightforwardly applicable at the current time.

5.3.15 Using the benchmark as a feature

Using the VPE-RES rankings as a feature gives the results in Table 5.21. The top scoring antecedent returned by VPE-RES is denoted by 1, the second 2, and so on. The aim is to see whether the ML algorithms will learn rules that complement VPE-RES. The results on the held-out data do not indicate that this is the case.

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
Exact Match	57.33	52.67	58.67	**60.67**	55.33
Head Match	73.33	70.00	78.00	**79.33**	73.33
Head Overlap	84.00	80.67	84.67	**86.00**	78.67
Miss	16.00	19.33	15.33	14.00	21.33

Table 5.21: Using the benchmark as a feature - Treebank data (Σ %)

5.3.16 Limiting the antecedent candidates

When the range of sentences used to look for potential antecedents was increased, this resulted in a large increase in the number of potential antecedents. To counteract the complexity that was added to the task, experiments were done to limit the search using clausal recency. Results are seen in Table 5.22. While some improvement is suggested, it is not definite, and this experiment is repeated during cross-validation testing (Subsection 5.3.19) to ensure important information isn't being removed. A range of 10 candidates each way seems to give a local optimum, and this will be used.

Clausal range	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
20	82.67	80.67	84.67	**86.00**	69.33
15	82.67	80.67	84.67	**86.67**	79.33
14	82.67	80.67	84.67	**86.67**	79.33
13	79.33	80.67	84.67	**86.00**	79.33
12	84.00	80.67	84.67	**85.33**	83.33
11	86.00	80.67	84.67	86.00	**87.33**
10	**87.33**	80.67	84.67	86.00	77.33
9	83.33	80.67	84.67	**86.00**	81.33
8	87.33	80.67	84.67	**86.00**	83.33
7	87.33	80.67	84.67	**86.00**	78.67
6	84.00	81.33	84.67	**86.00**	84.67
5	84.67	81.33	84.67	**86.67**	82.67
4	82.00	81.33	84.67	**86.00**	81.33
3	82.67	82.00	84.67	85.33	84.00
2	70.00	69.33	70.00	70.00	70.00

Table 5.22: Limiting the antecedent candidates (Head overlap Σ %)

5.3.17 Gain ratio of features

As in the VPE detection experiments, we examine the relative importance of the features as determined by Gain Ratio (Table 5.23)[6]. It is seen that SBAR and comparative relation are the most informative. Of the features added, coordination is seen to be highly informative as well. VPE-RES, while clearly very informative, gets a low ranking due to its large range of values.

5.3.18 Rules learnt by C4.5 and SLIPPER

The top rules learnt by the decision tree classifier are seen in Figure 5.6[7]. For the positive rules, it can be noticed that they all check for the *in quote* feature. Recency, in one form or another, is also found in all. Rule 117 checks for antecedents that are not in the same sentence as a VPE with a do-auxiliary, but still within a 6 word distance. Rule 45 checks for antecedents that are not the most recent, but are within a 4 word distance. Rule 52 incorporates the antecedent size feature.

[6]For the full table please see Appendix E.1.
[7]The full ruleset can be found in Appendix E.2.

```
                Positive :

Rule 37:
    Antecedent auxiliary = to
    Recency <= 1
    In-quotes = not_clashing
    VPE-RES rank <= 2
    ->  class TRUE  [95.6%]

Rule 45:
    Recency > 1
    Word distance <= 4
    In-quotes = not_clashing
    ->  class TRUE  [95.0%]

Rule 21:
        In-quotes = not_clashing
        Word distance <= 10
        VPE-RES rank <= 1
    ->  class TRUE  [88.6%]

Rule 117:
    Sentential distance > 0
    In-quotes = not_clashing
    VPE auxiliary = do
    Word distance <= 6
    ->  class TRUE  [95.5%]

Rule 52:
    In-quotes = not_clashing
    Antecedent size > 7
    Word distance <= 10
    VPE auxiliary = be
    VPE-RES rank <= 2
    ->  class TRUE  [92.2%]
```

```
                Negative :

Rule 181:
    Polarity = negative
    VPE-RES rank > 7
    VPE-RES rank <= 80
    ->  class FALSE  [99.9%]

Rule 175:
    VPE auxiliary = be
    VPE-RES rank > 7
    ->  class FALSE  [99.9%]

Rule 182:
        Adjuncts = no_adjunct
        VPE-RES rank > 7
        VPE-RES rank <= 80
    ->  class FALSE  [99.8%]

Rule 163:
    Sentential distance <= -1
    ->  class FALSE  [99.9%]

Rule 177:
    Word distance <= -3
    VPE auxiliary = do
    ->  class FALSE  [99.8%]
```

Figure 5.6: Top five positive/negative decision tree rules for antecedent location

Rank	Feature	Values	Gain Ratio
1	Comparative relation	2	0.32274332
2	SBAR relation	2	0.29467811
3	Coordination	2	0.14109133
4	Auxiliary match	2	0.021843147
5	In quotes	2	0.021253579
6	Be-do match	2	0.0082957198
7	Recency	20	0.049306709
8	Sentential distance	17	0.041288852
9	As-appositive	2	0.0021960193
10	Adjuncts	5	0.0035721699
11	Antecedent auxiliary	8	0.0030189935
12	VPE-RES rank	131	0.039990107
13	Word distance	259	0.034462515
14	VPE auxiliary	8	0.00036414261
15	Antecedent size	59	0.0021421686
16	Polarity	2	0.0000114919

Table 5.23: Contribution of antecedent features

In the negative rules, rule 181 states that any antecedents ranked between 7 and 80 by VPE-RES, and where the polarity is negative, are false. Rule 182 is similar to Rule 181, but checks for both VPs not having adjuncts instead. Rule 163 rules out all forward antecedents that are not in the same sentence as the VPE.

The top ten rules learnt by SLIPPER are seen in Figure 5.7[8]. The second and seventh rules learnt by SLIPPER work for antecedents occurring after the VPE, but are still in the same sentence (the contexts they were learnt from had 81 and 139 antecedents occurring before the VPE, respectively, so the antecedents occurring after the VPE had lower rank than these), and the third rule deals with antecedents occurring after the VPE as well. This shows that unlike decision trees, SLIPPER's boosting algorithm works to deal with these rare cases.

The top rule learnt deals with antecedents that are ranked fifth to seventh, but are within a 6 word distance. It is seen that some rather complicated rules are learnt, such as the eighth, which may suggest the need for a larger training corpus, as the rules are rather specific and suggest overfitting on few samples.

[8]The full ruleset can be found in Appendix E.3.

```
Default FALSE.
TRUE 0.028076 0 IF VPE-RES rank <= 7 Word distance <= 6 VPE-RES
    rank >= 5 Recency <= 7 .
TRUE 0.0277479 0 IF VPE-RES rank >= 82 Sentential distance >= 0
    VPE auxiliary = to .
TRUE 0.0214659 0 IF VPE-RES rank >= 60 Word distance >= -13 VPE-RES
    rank <= 60 Antecedent size <= 4 .
TRUE 0.0157358 0 IF Antecedent auxiliary = have Word distance <= 18
    Word distance >= -3 VPE-RES rank >= 2 VPE-RES rank <= 36 VPE-RES
    rank >= 6 Word distance <= 9 .
TRUE 0.0305465 6.50546e-05 IF Antecedent size >= 28
    Sentential distance >= 0 Word distance <= 6 VPE-RES rank <= 41 .
TRUE 0.0117885 3.68939e-05 IF Auxiliary match = aux_match
    Antecedent size <= 4 Antecedent size >= 2
    SBAR relation = not_sbar_rel VPE-RES rank <= 3
    Antecedent size <= 3  Recency >= 5 Sentential distance <= 2 .
TRUE 0.00640428 0 IF VPE-RES rank >= 140 Sentential distance >= 0 .
TRUE 0.0334545 0.000428514 IF VPE auxiliary = to Word distance <= 7
    Polarity = not_negative Recency >= -7 Adjuncts = ant_adjunct_only
    VPE-RES rank <= 46 VPE-RES rank >= 36 In-quotes = not_clashing .
TRUE 0.296091 0.00495414 IF VPE-RES rank <= 1 .
TRUE 0.00417111 2.68476e-05 IF Antecedent auxiliary = should
    Recency <= 2 VPE-RES rank <= 3 .
```

Figure 5.7: Top ten SLIPPER rules for antecedent location

5.3.19 Cross-validation

As cross-validation results cover all of the data, including both the training and test sets of the held-out data experiments, the benchmark was tested on this data as well, to ensure that performance stays similar, but the errors on the training data were not analyzed, as this might introduce bias into the testing procedure[9] (Table 5.24). The results are lower than just on the test set, but only by 4.57% for Head Overlap.

Several conclusions can be formed from the results of the cross-validation experiments (Table 5.25) :

- The integrated scoring scheme of VPE-RES performs better than ML us-

[9] A check for the performance of the coordination feature on the whole dataset was, however, performed, as it was seen only once in the test data. It is seen that this feature generates 6 true positives and 4 false positives. The errors it produces were not analyzed or corrected.

Criterion	#	%	Σ %
Exact Match	338	53.06	53.06
Head Match	90	14.13	67.19
Head Overlap	95	14.91	82.10
Miss	114	17.90	17.90

Table 5.24: New-VPE-RES performance on combined training and test corpus

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
Benchmark equivalent features only					
Exact Match	47.57	**50.08**	49.76	**50.08**	44.27
Head Match	61.70	63.74	**64.84**	64.52	58.08
Head Overlap	**79.91**	78.96	79.43	**79.91**	73.47
Miss	20.09	21.04	20.57	20.09	26.53
With extra features					
Exact Match	48.82	48.67	48.82	**53.38**	48.67
Head Match	62.17	62.17	63.27	**67.50**	61.54
Head Overlap	81.32	80.85	81.16	**82.89**	76.92
Miss	18.68	19.15	18.84	17.11	23.08
With VPE-RES feature, but without extra features					
Exact Match	51.18	52.43	53.22	**53.53**	52.75
Head Match	64.52	65.93	67.66	**67.82**	66.25
Head Overlap	82.57	82.57	83.67	**83.83**	82.10
Miss	17.43	17.43	16.33	16.17	17.90
With VPE-RES feature and extra features					
Exact Match	50.55	49.14	51.96	**54.47**	51.33
Head Match	65.31	62.64	66.25	**69.70**	65.62
Head Overlap	83.52	80.53	81.95	**85.24**	82.10
Miss	16.48	19.47	18.05	14.76	17.90
With antecedent distance limiting					
Exact Match	50.86	49.29	52.28	**54.79**	50.08
Head Match	65.31	62.95	66.88	**69.86**	64.68
Head Overlap	82.57	80.85	82.57	**85.87**	81.16
Miss	17.43	19.15	17.43	14.13	18.84

Table 5.25: Cross-validation results on Treebank (Σ %)

ing the same information. Even though the weights for the features are manually set, VPE-RES gives good results.

- With the addition of further features, ML results surpass VPE-RES.

- Unlike most of the original features, the contribution of the new features is not very high. Adding these to the VPE-RES system may result in a superior system, but might prove difficult to implement. This is because the weights for the features are manually determined for VPE-RES, and adding new features would require setting their weights in a way that would not unbalance the existing features. An ML algorithm such as genetic learning may be useful for the task of setting weights for existing and new features for VPE-RES while keeping its scoring architecture.

- With the addition of the VPE-RES rankings as well, ML results offer clear improvements.

- The new features are seen to offer consistent improvements, as they improve results even after VPE-RES is added.

- Clausal distance limiting is seen to offer improvements.

- Exact Match scores are improved by 1.73% over VPE-RES results, Head Match by 2.67% and Head Overlap by 3.77%, which corresponds to a 21% error reduction for Head Overlap.

- The held-out data size is too small to form conclusions for the antecedent location experiments as the benchmark, VPE-RES, already has significantly high performance.

5.4 Experiments using parsed data

For the parsed data, held-out experiments will not be used and only results on the whole dataset, or using cross-validation will be given.

5.4.1 Benchmark on parsed data

The benchmark algorithm can be applied as-is to data produced by Charniak's parser, as they use the same annotation scheme[10]. Table 5.26 shows the results of the benchmark applied to both parsed datasets . On the Treebank, the results are only slightly lower than those for manually annotated data. While there is a 12% degradation in Exact Match score, in Head Overlap the difference is only 5.61%. This shows that Charniak's parser is robust in producing the features used in the benchmark algorithm. Performance stays generally the same for the BNC data, and for the combined datasets as well.

	Treebank			BNC			Combined		
Criterion	#	%	Σ %	#	%	Σ %	#	%	Σ %
Exact Match	262	41.07	41.07	335	39.64	39.64	597	40.26	40.26
Head Match	128	20.06	61.13	112	13.25	52.90	240	16.18	56.44
Head Overlap	98	15.36	**76.49**	192	22.72	**75.62**	290	19.55	**75.99**
Miss	150	23.51	-	206	24.38	-	356	24.01	-

Table 5.26: New-VPE-RES performance on Charniak parsed data

Applying the benchmark algorithm to RASP data is more complicated, as equivalent category headers don't necessarily exist. The results (Table 5.27) are considerably lower than the Charniak equivalents. It is also seen that results for BNC are lower than for Treebank data using RASP. A significant source of error in RASP arises from fragmented sentences, such as the one seen in Figure 5.8. Here, because the elliptical verb 'will' is not parsed correctly, the VP 'think it' is chosen as the antecedent, despite containing the VPE.

Other errors arise from inconsistencies, such as the lack of a reliable analog to SBAR. RASP's tokeniser had problems with quotes, so these were stripped, meaning that the in-quote feature is not available. It is possible that a deeper analysis of RASP can give a better implementation and performance.

5.4.2 ML baseline

[10]While empty category information is not used for these experiments, the corpus is the same as in the previous chapter, including the added empty category information.

	Treebank			BNC			Combined		
Criterion	#	%	Σ %	#	%	Σ %	#	%	Σ %
Exact Match	187	29.31	29.31	180	21.30	21.30	367	24.75	24.75
Head Match	143	22.41	51.72	208	24.62	45.92	351	23.67	48.42
Head Overlap	84	13.17	**64.89**	97	11.48	**57.40**	181	12.20	**60.62**
Miss	224	35.11	-	360	42.60	-	584	39.38	-

Table 5.27: New-VPE-RES performance on RASP parsed data

```
(|T/frag| |Even:1_RR| |if:2_CS|
(|NP/plu1|
 (|N1/n1_nm| |baseball:3_NN1|
  (|N1/n1_pp1| (|N1/n1_nm| |trigger+s:4_NN2| (|N1/n| |loss+s:5_NN2|))
   (|PP/p1|
    (|P1/p_np| |at:6_II| (|NP/n1_name/-| (|N1/n| |CBS:7_NP1|)))))))
(|Tacl/dash-/-| |-:8_-|
 (|S/cj_end| |and:9_CC|
  (|S/np_vp| |he:10_PPHS1|
   (|V/do_bse/-| |do+s:11_VDZ|
    (|VP/not_vp| |not+:12_XX|
     (|V/np| |think:13_VV0| |it:14_PPH1|))))))
|will:15_VM|
...
```

Figure 5.8: Sentence fragment in RASP

Using the benchmark equivalent features on Charniak parsed data (Table 5.28), Exact Match score is degraded by 2.36%, and Head Overlap by 0.67% compared to VPE-RES. This difference is smaller than that seen for the Treebank data, which is likely due to a combination of two factors: the decrease in VPE-RES performance due to noise, and the robustness of the ML methods to this noise.

On RASP data (Table 5.29), the Exact Match score is identical to that of VPE-RES, and the Head Overlap score is only 0.07% reduced compared to VPE-RES.

5.4.3 Using all features

With the addition of the extra features and refinements discussed in previous experiments on the Treebank data, Exact Match score increases by 2.56% and Head Overlap by 2.36% for Charniak parsed data (Table 5.30). This is less of an

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
		LDC			
Exact Match	**39.81**	38.56	37.93	38.56	37.62
Head Match	**59.56**	57.52	57.52	57.52	55.96
Head Overlap	**75.71**	**75.71**	75.39	**75.71**	72.88
Miss	24.29	24.29	24.61	24.29	27.12
		BNC			
Exact Match	37.04	**37.16**	36.57	36.80	35.86
Head Match	49.23	**49.59**	49.35	49.23	48.28
Head Overlap	74.08	74.67	74.44	**74.79**	72.66
Miss	25.92	25.33	25.56	25.21	27.34
		Combined			
Exact Match	**37.90**	**37.90**	37.29	37.76	36.61
Head Match	52.87	**53.20**	53.07	53.00	51.45
Head Overlap	74.65	**75.32**	74.85	75.25	73.36
Miss	25.35	24.68	25.15	24.75	26.64

Table 5.28: Machine Learning performance on benchmark equivalent - Charniak parsed data - CV (Σ %)

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
		LDC			
Exact Match	27.43	29.00	**30.25**	29.00	26.33
Head Match	49.69	**50.78**	49.37	**50.78**	47.02
Head Overlap	62.85	**64.11**	63.17	**64.11**	60.03
Miss	37.15	35.89	36.83	35.89	39.97
		BNC			
Exact Match	20.12	**21.42**	21.18	**21.42**	14.20
Head Match	45.21	**45.68**	44.73	44.85	33.73
Head Overlap	56.92	57.40	57.51	**57.63**	41.89
Miss	43.08	42.60	42.49	42.37	58.11
		Combined			
Exact Match	22.86	**24.75**	24.54	24.68	22.45
Head Match	46.53	**47.54**	47.27	47.47	44.37
Head Overlap	59.00	**60.55**	60.28	60.42	56.84
Miss	41.00	39.45	39.72	39.58	43.16

Table 5.29: Machine Learning performance on benchmark equivalent - RASP parsed data - CV (Σ %)

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
		LDC			
Exact Match	39.97	38.24	40.60	**40.91**	35.74
Head Match	59.25	57.84	60.50	**62.23**	54.55
Head Overlap	76.02	75.39	76.80	**78.68**	70.38
Miss	23.98	24.61	23.20	21.32	29.62
		BNC			
Exact Match	36.21	34.44	39.29	**40.47**	34.32
Head Match	48.99	47.22	52.66	**54.20**	47.46
Head Overlap	74.32	73.96	75.50	**77.28**	68.64
Miss	25.68	26.04	24.50	22.72	31.36
		Combined			
Exact Match	37.83	35.81	39.99	**40.46**	37.76
Head Match	54.08	51.79	56.30	**57.32**	53.20
Head Overlap	75.79	73.84	76.40	**77.68**	72.89
Miss	24.21	26.16	23.60	22.32	27.11

Table 5.30: Machine Learning performance with extra features - Charniak parsed data - CV (Σ %)

Criterion	Tree	MBL	GIS-ME	L-BFGS-ME	SLIPPER
		LDC			
Exact Match	24.14	25.86	**28.68**	25.71	21.63
Head Match	46.71	46.39	49.53	**50.63**	44.36
Head Overlap	64.89	62.38	63.95	**67.55**	61.60
Miss	35.11	37.62	36.05	32.45	38.40
		BNC			
Exact Match	18.46	19.88	**20.95**	19.05	15.86
Head Match	41.66	39.76	**45.44**	44.50	37.87
Head Overlap	57.75	54.20	57.51	**59.41**	51.72
Miss	42.25	45.80	42.49	40.59	48.28
		Combined			
Exact Match	20.57	21.78	**24.21**	22.18	18.21
Head Match	43.02	42.35	**47.34**	46.80	40.93
Head Overlap	59.74	58.66	60.35	**62.58**	56.57
Miss	40.26	41.34	39.65	37.42	43.43

Table 5.31: Machine Learning performance on with extra features - RASP parsed data - CV (Σ %)

improvement than was seen on the Treebank data, which suggests that the new features were not extracted as successfully on the parsed data. On RASP parsed data, Exact Match is reduced by 0.54%, and Head Overlap increased by 2.03%, which suggests that the extra features were even less successful here.

5.5 Summary of Chapter

This chapter presented a high-performance, robust antecedent location system. First, evaluation of an existing method for VPE antecedent location was performed. This was followed by application of its components to machine learning. Finally, new features were added to the ML classifiers. To summarize the results:

- Experiments utilising five different machine learning algorithms (decision trees, MBL, GIS-based and L-BFGS-based maximum entropy modelling, SLIPPER) have been conducted on the task of locating VPE antecedents. Decision trees, which were not suitable for the VPE detection experiments (see section 4.2.4), are used here, as the data is more balanced and useful results are obtained.

- Using a re-implementation of Hardt's (1997) VPE-RES algorithm on the Penn Treebank data, for which it was designed, 82% Head Overlap (HO) is obtained.

- Using the preference factors of VPE-RES as features for machine learning gives 80% HO. Adding further syntactic features improves results to 83% HO. Adding the VPE-RES results as a feature as well, final performance on the Treebank is 86% HO using ML techniques. These results show clear improvement over the VPE-RES results, and represent a 21% reduction in error.

- As in the VPE detection experiments, two parsers were used, Charniak's and RASP. Re-parsing the Treebank, VPE-RES achieves 76% HO using Charniak parsed data, and 65% using RASP parsed data. As Charniak's parser uses the same representation as the Treebank, VPE-RES does not need to be modified for it. RASP uses a different representation, and it is

seen that it is not possible to reliably translate all parts of the algorithm. The results on RASP are significantly affected by errors introduced in the parsing process. Using ML on the re-parsed Treebank data, top scores are 79% for Charniak parsed data and 68% for RASP parsed data.

- Parsing the BNC, VPE-RES achieves 76% HO using Charniak parsed data, and 57% using RASP parsed data. Results using Charniak's parser are seen to be very close to those on the re-parsed Treebank, even though the parser is not trained on the BNC. RASP results, on the other hand, are noticeably degraded. Using ML on the parsed BNC data, top scores are 77% for Charniak parsed data and 59% for RASP parsed data.

- Combining the datasets, VPE-RES gets 76% HO on Charniak parsed data, and 61% on RASP parsed data. Using ML on the combined dataset parsed using Charniak's parser, the highest result is 78% HO, and using RASP, 63%. This gives the final result for the experiments, with 78% HO possible using parsed data.

- While results are high for Head Overlap, Exact Match, scores are around 55% for Treebank data, and 40% for parsed data. This can present a problem for subsequent NLP modules, such as resolution, described in the next chapter, as it is Exact Match that is required to produce intelligible sentences in a portion of the cases. This suggests that further work needs to be done on improving the Exact Match score.

- It is seen that in all instances using ML and the new features results in an improvement over VPE-RES alone. The improvements are most noticeable on the Treebank data, but are clear on parsed data as well.

- Decision trees give average performance, and can be used successfully on the antecedent location dataset (these results, and all those following, are for Head Overlap).

Corpus	ML Baseline	+ extra features
Treebank	79.91	82.57
Charniak-Treebank	75.71	76.02
RASP-Treebank	62.85	64.89
Charniak-BNC	74.08	74.32
RASP-BNC	56.92	57.75
Charniak-Combined	74.65	75.79
RASP-Combined	59.00	59.74

- MBL gives average performance, and is particularly useful when dealing with fewer features and on parsed data.

Corpus	ML Baseline	+ extra features
Treebank	78.96	80.85
Charniak-Treebank	75.71	75.39
RASP-Treebank	64.11	62.38
Charniak-BNC	74.67	73.96
RASP-BNC	57.40	54.20
Charniak-Combined	75.32	73.84
RASP-Combined	60.55	58.66

- GIS-MaxEnt gives average performance, with its best results on RASP parsed data.

Corpus	ML Baseline	+ extra features
Treebank	79.43	82.57
Charniak-Treebank	75.39	76.80
RASP-Treebank	63.17	63.95
Charniak-BNC	74.44	75.50
RASP-BNC	57.51	57.51
Charniak-Combined	74.85	76.40
RASP-Combined	60.28	60.35

- L-BFGS-MaxEnt is still generally the best performing classifier, but less clearly so than on the VPE detection experiments. It takes the lead especially on the original Treebank data, and with higher numbers of features.

Corpus	ML Baseline	+ extra features
Treebank	79.91	85.87
Charniak-Treebank	75.71	78.68
RASP-Treebank	64.11	67.55
Charniak-BNC	74.79	77.28
RASP-BNC	57.63	59.41
Charniak-Combined	75.25	77.68
RASP-Combined	60.42	62.58

- SLIPPER is again consistently the worst performing classifier, generally outperformed by decision trees too, the only other rule-based classifier in the experiments. This may suggest the need to weight positive examples as in the VPE detection experiments.

Corpus	ML Baseline	+ extra features
Treebank	73.47	81.16
Charniak-Treebank	72.88	70.38
RASP-Treebank	60.03	61.60
Charniak-BNC	72.66	68.64
RASP-BNC	41.89	51.72
Charniak-Combined	73.36	72.89
RASP-Combined	56.84	56.57

The results show that it is possible to successfully identify VPE antecedents, even using automatically parsed data. Consistent improvements are made over previous work. This is also the first work to examine performance on parsed data for the task, but admittedly the application is straightforward.

Many of the features used for this task can be informative for other forms of ellipsis as well, including recency, SBAR-relation, comparative relation, and coordination. This provides a good framework for future work into other forms of ellipsis resolution.

Further work can be done into providing a robust subject matching feature, as discussed in section 5.3.14. A robust dialogue tracking system would also allow for useful features to be extracted. Work has been done (Hardt, 1998) on improving

the Exact Match score using ML techniques on the identified antecedents. This can be used and be improved upon.

Chapter 6

Resolving the antecedent

The work in the previous two chapters identifies VPE sites and finds the antecedent VPs for VPEs. This data needs to be resolved before it can be used in further processing, which is addressed in this chapter.

Section 6.1 provides a description of the approach that will be taken in this chapter and its aims.

Section 6.2 describes relevant previous work.

Sections 6.3, 6.4 and 6.5 enumerate types of VPE classified as trivial, intermediate and difficult, respectively.

Section 6.6 describes experiments on the Penn Treebank and parsed data on automatic classification and resolution of sentences.

6.1 Overview

The aim of this chapter is not to argue for any particular approach to VPE resolution, but simply to categorize the instances in the corpora used according to the complexity required to deal with them. The measure used to determine this will be readability, i.e. whether the sentence is sensible after reconstruction. Phenomena such as anaphoric dependencies and traces are beyond the scope of this module, and need to be dealt with elsewhere.

As syntactic reconstruction is a simpler process than semantic approaches, this will be taken as the baseline. The type of syntactic approach being used as a baseline will be described in the cases it needs to deal with, and is simple enough so that if desired it can be implemented for practical use with ease. Semantic resolution systems, on the other hand, are theoretically capable of dealing with most cases in the data, but building robust semantic resolution systems would require a significant investment of effort.

In a number of cases, straightforward copying of the antecedent VP to the VPE site will result in an intelligible sentence. For a variety of constructions this is not enough, and further processing needs to be done. Of these cases, some can be dealt with using simple transformations, but the rest require more elaborate methods.

This chapter attempts to provide a classification of the data in the corpora used according to these categories. Statistics were collected from all the corpora used on how often these categories occur. It should be noted that for a single case of VPE more than one category may apply, so adding the percentages quoted would result in a number greater than 100%.

Based on those instance classified as 'trivial', a syntactic resolution model is also implemented, and its output is checked against human annotated data to assess how much of the data can be reliably resolved using a simple syntactic approach.

6.2 Previous work

The discussion of the literature concerned with this phase of ellipsis resolution can be found in Section 2.1. There is a large literature that discusses the level at which resolution occurs, the general processes involved, and how these interact with difficult cases and other linguistic phenomena. Generally, the approaches presented are not actually implemented or tested on corpora, although their application is made clear.

A notable exception is the generalized reconstruction system described in (Lappin and Shih, 1996; Gregory and Lappin, 1997), which is designed for a variety of ellipsis types. While this algorithm has been implemented, results from tests on

corpora are not available.

6.3 Trivial cases

The baseline approach will cover the cases where resolving the antecedent is trivial, and resolution can be successfully done by copying the antecedent to the VPE site, or by the use of simple rules that depend on the syntactic context. Overall, 83.8% of the cases of VPE in the data belong only to one or more of the categories outlined below, showing that the majority of VPE instances pose a comparatively simple resolution task.

6.3.1 Simple copy

For 44.4% of the VPE instances found, straightforward copying is sufficient to produce a readable sentence.

(46) Jewelry makers rarely pay commissions and aren't expected to anytime soon. [pay commissions]

Cases involving negative markers in the antecedent that need to be removed before copying to the VPE site are included in this category.

(47) In 1893 while recovering from a bout of influenza he wrote : You will see from this heading that I am not dead yet, nor likely to be. [dead yet]

If the VPE is followed by a negation, the antecedent is copied after this.

(48) a. It does not mean he is mad.

b. It does not mean he isn't. [mad]

6.3.2 Replace

Appending the antecedent to the VPE site, as seen in Simple Copy, is taken as the default, but some auxiliaries (doing / done / to do) need to be removed during resolution[1]. This covers 1.6% of cases in the data.

(49) a. Such honesty brought him more trouble than hypocrisy would have done - with Louise Colet, for instance.

b. * Such honesty brought him more trouble than hypocrisy would have done *brought him* - with Louise Colet, for instance.

c. Such honesty brought him more trouble than hypocrisy would have *brought him* - with Louise Colet, for instance.

6.3.3 Tense

For 21.3% of cases, the tense of the antecedent needs to be adjusted to that of the VPE site.

(50) a. You said something - about getting caught up in the action -

b. PLAYER : (Gaily freeing himself) I did, I did, - You're quicker than your friend . [say something about getting caught up in the action]

6.3.4 Questions

In 21.4% of cases, where the VPE is part of a question, the antecedent needs to be placed before the end of the sentential unit.

(51) We got his symptoms, didn't we ? [get his symptoms]

(52) "It's going to be a long time before we get another chance, though, isn't it though ?" [going to be a long time before we get another chance]

[1]See Appendix F for a discussion on this.

6.3.5 Neither / nor

As with Questions, in 1.2% of cases, where the VPE occurs in a neither/nor construction, the antecedent needs to be placed before the end of the sentential unit.

(53) a. Do you ever think of yourself as actually dead, lying in a box with a lid on it ?

b. GUIL : No.

c. ROS : Nor do I, really. [think of myself as actually dead]

6.4 Intermediate cases

The label "intermediate cases" applies to those where it is possible to generate rules for resolution, but in a less straightforward way than for the trivial cases. The correct resolution of these cases is also more open to interpretation. Intermediate cases occur in about 8% of the corpus.

6.4.1 'As' appositives

VPEs that occur immediately following an *as* require the antecedent to be placed after the clause following the VPE, as seen in (54b). Depending on personal preference, a small change in structure for readability may also be appropriate (54c). These cases occur 2.9% of the time.

(54) a. The party divisions between leading reformers such as Lloyd George, Mosley and Macmillan also hindered co-operation, as did personal hostility within political parties (on all occasions between Mosley and Ernest Bevin in the Labour movement and intermittently between Lloyd George and Keynes in the Liberals).

b. The party divisions between leading reformers such as Lloyd George, Mosley and Macmillan also hindered co-operation, as personal hostility

within political parties (on all occasions between Mosley and Ernest Bevin in the Labour movement and intermittently between Lloyd George and Keynes in the Liberals) *hindered co-operation.*

c. The party divisions between leading reformers such as Lloyd George, Mosley and Macmillan also hindered co-operation, *and* personal hostility within political parties (on all occasions between Mosley and Ernest Bevin in the Labour movement and intermittently between Lloyd George and Keynes in the Liberals) hindered co-operation *too.*

6.4.2 'So' anaphora

In 3% of cases, the VPE is preceded by *so* (and in one case, *also*). These cases require a reversal of structure to mirror the antecedent. Kehler only considers *do so/so doing* as anaphoric, but it is seen that *so* constructions with any type of auxiliary require more than simple reconstruction.

(55) a. I 'm afraid.

b. So am I. [I am afraid too]

(56) a. "I'm still alive, Fiver" he said.

b. "So are all of us."

c. *"All of us are still alive [too]"*

6.4.3 Determiners

Any determiners in the antecedent have to be modified to reflect the VPE site. 1.5% of the antecedents contain determiners.

(57) I still kind of don't believe it, Cale mumbles into his sleeve, although modesty is not one of his foremost traits; frankness, bravery and literacy are. [some of his foremost traits]

6.4.4 Which anaphora

VPE sites that have *which* as their subject require the resolution of this anaphor, which happens in 0.4% of cases[2].

(59) a. If he didn't talk sense, which he does.

b. If he didn't talk sense, *but* he does *talk sense*.

(60) a. ROS : He wouldn't discriminate between us.

b. Even if he could.

c. Which he never could.

d. *But* he never could *discriminate between us*.

6.4.5 Comparatives

VPEs that occur in certain comparative constructions require the antecedent to be placed after the clause following the VPE. There is one case of this kind.

(61) a. The dominance of orthodox economic policies, cautious pragmatism and consensus politics reflected the mood of the British electorate in the 1920s rather more than did the utopian assumptions of the Die-hard remnant.

b. The dominance of orthodox economic policies, cautious pragmatism and consensus politics reflected the mood of the British electorate in the 1920s rather more than the utopian assumptions of the Die-hard remnant *reflected the mood of the British electorate*.

[2]Although the examples in the corpus, such as (59) and (60), don't cover it, it is also possible to have examples such as (58).

(58) a. If he can get there.

b. Which he can.

c. *And* he can *get there*.

6.4.6 Chained VPE

Three cases in the data (0.2%) require information from an antecedent which is a resolved VPE itself.

(62) a. Your kind of thing, is it ? [your kind of thing]

b. Well, no, I can't say it is, really. [my kind of thing]

This is not difficult to resolve in itself, but differs from the approach that was taken in this work.

6.5 Difficult cases

A number of cases, while trivial for humans to resolve, pose significant problems for automatic methods of ellipsis resolution. These cases cover 8.2% of the data.

6.5.1 Pronominal ambiguity

Examining the VPEs, it is seen that there are 313 cases where the antecedent contains a pronominal, which is around 22% of the samples. In terms of ambiguity these cases can be classified as follows :

- In 228 cases resolving the copied pronominal to the pronominal it was copied from (the strict reading) is sufficient.

  (63) a. "Have you$_i$ seen him$_j$?"

  b. Now that Hazel$_i$ thought about it, he$_i$ had not. [seen him$_j$]

- In 70 cases, resolving to the original pronominal is still correct, but the syntactic form needs to change to reflect the change in speaker. A discourse tracking system is required for these cases.

  (64) a. "Take care of yourself$_i$ then"

b. "I$_i$ will." [take care of myself$_i$]

(65) a. "I ain't going to fight you$_i$ no more".

b. "I know you ain't", Dan affirmed, feeling ten feet tall. [going to fight me$_i$]

- In 15 cases, ambiguity results from the syntactic copying, and sloppy readings may be necessary. These cases are more difficult and require the generation of all possible readings for further processing by other NLP modules, and if possible, the selection of the appropriate one based on contextual information.

(66) a. Now Hans had given Ma something of his - we both had when we thought she was going straight to Pa... [given Ma something of his/ours]

b. If I don't put my two cents in soon, somebody else will. [put my/their two cents in soon]

c. I am he as he is me, as we are all. [he/me/ourselves]

It is seen that pronouns in the antecedent which need processing happen with enough frequency (5.8% of the corpus) that whenever they are encountered a special module should be utilised. The number of cases where ambiguity resolution is needed, however, is quite small, and constitutes about 1% of the corpus, and 5% of cases with pronominals in the antecedent.

6.5.2 Cases requiring inference

In 1% of cases, information from non-syntactic sources seems necessary. Some of the more interesting examples encountered are as follows.

(67) a. "He's got this idea about drying out."

b. "It ain't an idea !"

c. "If it ain't an idea", she said, "how comes it you can drink beer but not water ?"

d. He looked piously to heaven and said, "Beer don't affect the tissues none", and the ingenious hypocrisy of this defense pleased Henrietta so that she forgave him his stint of malevolence.

e. His grand-daughter sighed.

f. "Come on, do." [drink water]

(68) a. "Fancy a trip to the theatre ?"

b. The question was close to being a statement.

c. "I'd - I'd love to", I said. [go to the theatre]

(69) a. "I could get anyone under the spell", he says, adding that he had hypnotised their maid (as Breavman had in The Favourite Game) and feared that he had driven her insane by it ! [hypnotised someone]

In the examples above, the antecedents are never explicitly uttered, but can be extracted from the existing data.

6.5.3 Trace in antecedent

In 0.7% of cases, the antecedent contains a trace which needs to be resolved.

(70) Such genuine human leadership the proprietorship can offer *t*, corporations can not. [offer such genuine human leadership]

(71) a. This isn't what I should feel *t* for him.

b. But I do. [feel this for him]

(72) Again there was something familiar about her, something – "You haven't got cancer", I said *t* as strongly as I could. [say "You haven't got cancer"]

6.5.4 Unspoken antecedents

0.3% of the VPEs in the corpora refer to an unspoken VP antecedent.

(73) a. Cigarette ?

b. No, I didn't think you would. [smoke/want a cigarette]

c. You don't mind if I do ? [smoke a cigarette]

(74) a. "How about tonight", she said and the pathos in her ignorant unknowing enquiry almost made me gag.

b. My mind flew back to the sight of The Fat Controller's cigar.

c. I had trodden on its shattered corpse that morning on my way out of the caravan.

d. "I - I can't, really, not tonight. [be with you]

6.5.5 Split antecedents

Split antecedents occur in 0.3% of cases, where syntactic data from separate phrases, even sentences have to be assembled.

(75) a. You never thought that being grown up would mean having to be quite so - how can I put it ?

b. Quite so - grown up.

c. Now did you ? [ever think that being grown up would mean having to be quite so grown up]

(76) And besides, you do not look, you do not choose, do you ? [look and/or choose]

6.5.6 Nominalization

Three cases in the data have nominalized antecedents.

(77) a. "Trust."

b. "Yes."

c. That's what he didn't, the water here. [trust]

(78) Even the knowledge that she was losing another boy, as a mother always does when a marriage is made, did not prevent her from having the first carefree, dreamless sleep that she had known since they dropped down the canyon and into Bear Valley, way, way back there when they were crossing those other mountains. [know]

6.6 Building a simple resolver

To demonstrate the feasibility of building a resolution system based on the patterns seen in the data, a simple rule-based system was implemented. This requires two stages; the first detecting the patterns, and the second applying the transformations associated with them.

The detection part uses some very simple searches to detect classes, with some examples below :

- The Tense checks are performed using the VPE auxiliary and the head verb of the antecedent. The list of applicable tenses following a particular auxiliary are limited (*is* and *doing* must be followed by a present participle, *do, does, to, will, might, could* must be followed by a base verb or a present tense, non-3rd person singular verb, etc.). Any mismatches are flagged as requiring a tense shift.

- Questions are detected by looking for VPEs followed optionally by a negation and a pronominal, and ending with a question mark.

- Pronominals and Traces are detected straightforwardly by determining if any syntactic units identified as pronominals or traces are present in the antecedent clause.
- 'As' appositives are detected by the presence of the word 'as' before the antecedent, and likewise for 'So' and 'Which' anaphora, and 'Neither/nor' cases.
- Anything not flagged by any of the other detectors is marked as a Simple copy.
- Some of the difficult cases, as well as chained VPEs, are not dealt with.

6.6.1 Treebank data

Using these checks, the results in Table 6.1 are obtained. Each row shows the performance of the heuristic designed to classify that category. The 'weighted average' here is obtained by summing over all the True Positive and False Positive counts for all categories.

Class	No of cases	Recall	Precision	F1
Simple copy	298	72.82	97.31	83.30
Replace	14	100.00	100.00	100.00
Tense	169	96.45	82.74	89.07
Question	81	97.53	85.87	91.33
Neither / nor	9	100.00	100.00	100.00
As appositive	21	100.00	100.00	100.00
So anaphora	26	96.15	92.59	94.34
Determiner	9	100.00	27.27	42.86
Which anaphora	1	100.00	100.00	100.00
Comparative	0	-	-	-
Pronominal	36	91.67	27.50	42.31
Trace	5	80.00	7.27	13.33
Weighted average	**655**	**85.65**	**72.11**	**78.30**

Table 6.1: Antecedent resolution classification - Treebank data

Some of the errors introduced are due to incorrect tags in the Treebank, while others require improvements on the detection checks. Most of the errors come from low precision checks which produce large numbers of false positives, namely Determiner, Pronominal and Trace.

A proof-of-concept system was also built to apply the necessary transformations for the simple cases that were detected as being such. The Simple copy, Replace and Question modules were implemented, while the Tense module identifies the verb that needs changing and the tense to change it to, but is not connected to a lexicon to look it up. This system gets 61.47% of the resolved sentences correct.

To improve results, the low precision checks (Determiner, Pronominal and Trace) were not performed. While this of course gives zero performance for those categories, the overall result is an increase in performance, as these categories generate many False Positive errors. This improves the Simple copy check to 92.73% F1[3], and the weighted average to 88.44% F1[4]. The rate of successful resolution also increases to 80.97%. Implementing the intermediate modules, which would bring the score into the 90% range, is left for future work.

6.6.2 Parsed data

For the parsed data experiments only Charniak parsed data will be looked at, as traces need to be detected and the RASP parsed data does not contain this.

Class	No of cases	Recall	Precision	F1
Simple copy	298	76.09	96.17	84.96
Replace	14	100.00	100.00	100.00
Tense	169	95.88	83.16	89.07
Question	81	96.30	85.71	90.70
Neither / nor	9	100.00	100.00	100.00
As appositive	21	100.00	100.00	100.00
So anaphora	26	88.46	92.00	90.20
Determiner	9	100.00	27.27	42.86
Which anaphora	1	100.00	100.00	100.00
Comparative	0	-	-	-
Pronominal	36	91.67	27.50	42.31
Trace	5	0	0	0
Weighted average	**655**	**85.95**	**75.07**	**80.14**

Table 6.2: Antecedent resolution classification - Charniak parsed Treebank data

Interestingly, on the Treebank data, the parsed data gets better results than the original Treebank data (Table 6.2), but this is only due to fewer false pos-

[3] 96.31% recall and 89.41% precision
[4] 89.31% recall and 87.57% precision

itives being returned by the Trace check. 64.15% of sentences are correctly resolved. Removing the low precision checks, the weighted average for classification is 87.90%[5]. The rate of successful resolution increases to 80.81%.

Class	No of cases	Recall	Precision	F1
Simple copy	647	75.89	93.52	83.79
Replace	23	100.00	100.00	100.00
Tense	310	95.81	78.16	86.09
Question	311	96.46	88.50	92.31
Neither / nor	17	100.00	100.00	100.00
As appositive	42	100.00	97.67	98.82
So anaphora	43	86.05	92.50	89.16
Determiner	22	90.91	30.30	45.45
Which anaphora	6	66.67	100.00	80.00
Comparative	1	0	0	0
Pronominal	85	90.59	24.76	38.89
Trace	10	10.00	1.96	3.28
Weighted average	**1493**	**86.14**	**72.41**	**78.68**

Table 6.3: Antecedent resolution classification - Charniak parsed Treebank and BNC data

On the combined dataset, results remain similar (Table 6.3). The rate of successfully resolved sentences is 62.18%. Removing low precision checks, the weighted average for classification is increased to 86.51%F1[6], and 78.66% of sentences are correctly resolved. Parsing errors introduce some errors, but not many. Given that the maximum coverage that could have been achieved by resolving trivial cases is 83.8% , the rate of error is quite low.

6.7 Summary of Chapter

The main contribution of this chapter is in providing a classification and statistical assessment of the types of ellipsis by resolution complexity. The 'complexity' of the cases of course differs according to the resolution methodology adopted, but the judgement offered here takes straightforward syntactic manipulation to be 'trivial', and cases which would be claimed to give support to semantic approaches to be 'difficult'.

[5] 88.70% recall and 87.11% precision
[6] 88.08% recall and 85.00% precision

What the analysis of the corpus shows is that 83.8% of VPE cases are trivial. A further 8% are of intermediate difficulty and can mostly be handled using syntactic transformations. The remaining 8.2% are the more difficult cases. This shows that while simple syntactic reconstruction can cover more than 90% of cases, for the remaining cases further processing is necessary.

The majority of these difficult cases are due to pronominal ambiguity, and there are ways of generating at least all the possible readings in such cases. Only 2.3% of the data[7] cannot be detected using syntactic checks (given correct data) and constitute the cases usually argued to motivate semantic approaches. If strict identity is assumed for pronominals by default, and dialogue tracking is used to handle speaker changes, all of the data except for 3.3%[8] can be handled using simple syntactic reconstruction methods. Without a dialogue tracking system, the number of cases that can't be handled correctly is 8.1%.

The high percentage of cases which can be resolved using 'cheaper' syntactic methods, coupled with a percentage of cases which require more complex methods that, while small, is not insignificant, confirms that from an engineering perspective, high-coverage systems can be built without the use of more complex systems such as semantic resolution[9].

The application aspect of resolution was not given the same weight as the application of VPE detection and antecedent location due to the fact that there already exists a large literature on this topic. The difficult cases require complex approaches that are beyond the scope of this work, while the simpler cases can be implemented using simple rule-based systems.

A prototype system was built to demonstrate these claims, which was shown to

[7]Cases of Inference, Unuttered antecedents, Split antecedents and Nominalization.

[8]Cases of Inference, Unuttered antecedents, Split antecedents, Nominalization and ambiguous pronominals.

[9]The high percentage of simple cases also confirms the viability of composite models, such as the sequenced model proposed by Lappin (2004), where syntactic salience and recency measures are used to handle the majority of anaphora and ellipsis resolution tasks. Those cases that cannot be resolved using this first step can be dealt with using statistically determined lexical preference involving semantic and pragmatic knowledge. Finally, cases that cannot be dealt with using either of these two approaches will have to be dealt with using inference.

If appropriate confidence measures can be employed such that each module is dealing with those instances which should be assigned to it, a sequenced model has the advantage of dealing with instances of ellipsis using the computationally cheapest and most robust approach possible.

work with 88% F1 for classification and to generate readable resolved sentences for 81% of the test corpus. Tested on parsed data, classification works equally well, with very little degradation. On the whole parsed dataset, 86.5% F1 is achieved for classification, with 79% coverage. These numbers can be improved on, as the system built is meant as a proof of concept.

Chapter 7

Conclusions

This work has presented a corpus-based approach to the study and processing of Verb Phrase Ellipsis. The system developed identifies elliptical verbs, finds the appropriate antecedent, and chooses between different possible resolutions, depending on the context.

All of these stages are used to investigate the empirical aspect of VPE, and the success of different approaches. Machine learning techniques are used to optimize results where applicable, and to ensure that the methods are kept non-specific to allow easy application to other domains.

This work offers the first focused, large scale work on the topic of VPE detection. Using only POS tags and lexical form information, 76% F1 is obtained on parts of the BNC dataset. On the Penn Treebank dataset, 82% F1 is obtained using further syntactic features. This offers an improvement over previous results on the same data, which achieve 48% F1. Using an automatically parsed version of the Treebank dataset gives 67% F1, and when this is combined with similarly parsed BNC data, 71% F1 is obtained overall.

For the task of antecedent location, existing work is built upon and extended, and used on parsed data for the first time. On Treebank data, 86% Head Overlap is achieved between the antecedent found and the real antecedent. This improves on previous results of 82% on the same data. On the parsed Treebank 79% is achieved, and combining this with the BNC data, the final score for parsed data is 78%.

The chapter on VPE resolution offers a corpus-based attempt at discerning how often different types of VPE occur, and it is the first to attempt this with resolution complexity as a measure, to the best of my knowledge. Ellipses are divided into three broad categories : *trivial*, which can be resolved using very simple syntactic reconstruction; *intermediate*, which can still be resolved using syntactic reconstruction, but less straightforwardly; and *difficult*, which require more complex methods than simple reconstruction. In each of these categories several subcategories exist, identified by syntactic conditions.

This categorisation of the data shows that 84% of cases are trivial, 8% intermediate, and 8% difficult. Of these, only 2.3% are difficult to detect, and belong to classes argued to require semantic resolution systems, while the rest can be marked as potentially requiring further processing by other modules. Furthermore, if a simple syntactic reconstruction approach is taken, coupled with a dialogue tracking system, only 3.3% of the data cannot be handled, giving credibility to the use of syntactic approaches to VPE resolution from an engineering perspective.

A simple rule-based system was built to perform this classification, and to generate sentences for the trivial cases. Classification is performed with 88% F1, and all trivial cases which are detected are generated correctly, giving correct readings for 81% of cases. On parsed data, 86.5% F1 is achieved for classification, with 79% of cases in the data correctly resolved.

The central claim for this work was that it is possible to build methods for each stage of VPE resolution using simple, robust systems. This has been shown to be true, with good results at each stage, whether on manually annotated data or parsed data. It has been shown that each stage of the work can be performed with high coverage using only simple syntactic data, and that the cases requiring complex methods are a small minority of the data. Given these results, it is shown that a useable, stand-alone VPE resolution system is implementable.

Directions for future work have been outlined for each section in their respective chapter summaries. The most obvious one is connecting the individual stages into a single system. This has not been done for this work, because while each stage performs well individually, errors would propagate through the stages, resulting in a noticeable level of errors at the complete end. A sizeable portion of the an-

tecedents found are not exact matches, and would require cleaning up to produce readable sentences. The modifications necessary to chain the stages together are not, in principle, difficult to achieve, but these are engineering challenges for future work. This does not detract from the viability of the approach, nor does it have theoretical implications. It simply indicates that real-world applications require, as usual, some adjustments.

References

C. Aone and S. W. Bennet. 1996. Applying machine learning to anaphora resolution. In S. Wernter, E. Rillof, and G. Scheler, editors, *Connectionist, statistical and symbolic approaches to learning for Natural Language Processing*, pages 302–314. Springer-Verlag, Berlin.

Michele Banko and Eric Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *ACL*.

Adam Berger, Stephen Della Pietra, and Vincent Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1).

Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. 1984. *Classification and Regression Trees*. Wadsworth, Belmont, CA.

Leo Breiman. 1996a. Bagging predictors. *Machine Learning*, 24(2):123–140.

Leo Breiman. 1996b. Bias, variance, and arcing classifiers. Technical Report 460, Department of Statistics, University of California at Berkeley.

Eric Brill and Philip Resnik. 1994. A rule-based approach to prepositional phrase attachment disambiguation. In *Proceedings of COLING'94*.

Eric Brill. 1992. A simple rule-based part of speech tagger. In *Proceedings of the Third ACL Applied NLP*.

Eric Brill. 1993a. Automatic grammar induction and parsing free text: A transformation-based approach. In *Proceedings of ACL*.

Eric Brill. 1993b. *A Corpus-Based Approach to Language Learning*. Ph.D. thesis, University of Pennsylvania.

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.

Eric Brill. 1996. Learning to parse with transformations. In *Recent Advances in Parsing Technology*. Kluwer Academic Publishers.

Eric Brill. 1997. Unsupervised learning of disambiguation rules for part of speech tagging. In *Natural Language Processing Using Very Large Corpora*. Kluwer Academic Publishers.

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.

Richard Campbell. 2004. Using linguistic principles to recover empty categories. In *Proceedings of the 42nd annual meeting of the Association for Compuatational Linguistics*, pages 646–653.

Claire Cardie. 1993. Using decision trees to improve case-based learning. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 25–32.

B. Cestnik, I. Konenenko, and I. Bratko. 1987. Ssistant 86: A knowledge elicitation tool for sophisticated users. In I. Bratko and N. Navrac, editors, *Progress in Machine Learning*. Sigma Press, UK.

Wynn Chao. 1987. *On Ellipsis*. Ph.D. thesis, University of Massachusetts at Amherst.

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

Stanley F. Chen and Joshua Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting of the Association for Compuatational Linguistics*, pages 310–318.

S. Chen and R. Rosenfeld. 1999. A gaussian prior for smoothing maximum entropy models. Technical report, Carnegie Mellon University.

Noam Chomsky. 1981. *Lectures on Government and Binding*. Foris Publications, Dordrecht.

Noam Chomsky. 1982. *Noam Chomsky on the Generative Enterprise*. Foris Publications, Dordrecht.

William W. Cohen and Yoram Singer. 1999. A simple, fast, and effective rule learner. In *Proceedings of the 16th National Conference on AI*.

William W. Cohen. 1995. Fast effective rule induction. In *Machine Learning: Proceedings of the Twelfth International Conference*.

William W. Cohen. 1996. Learning rules that classify e-mail. In *AAAI Spring Symposium on ML and IR.*

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

S. Corston-Oliver. 2000. Using decision trees to select the grammatical relation of a noun phrase. In *Proceedings of the 1st SIGDial workshop on discourse and dialogue*, pages 66–73.

N. Cristianini and J. Shawe-Taylor. 2000. *An introduction to Support Vector Machines (and other kernel-based learning methods).* Cambridge University Press.

Walter Daelemans, Jakub Zavrel, Peter Berck, and Steven Gillis. 1996. Mbt: A memory-based part of speech tagger-generator. In *Proceedings of the Fourth Workshop on Very Large Corpora*, pages 14–27.

Walter Daelemans, Sabine Buchholz, and PJorn Veenstra. 1999a. Memory-based shallow parsing. In *Proceedings of CoNLL-99.*

Walter Daelemans, Antal van den Bosch, and Jakub Zavrel. 1999b. Forgetting exceptions is harmful in language learning. *Machine Learning, special issue on natural language learning*, 34:11–43.

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.

Walter Daelemans, editor. 1999. *Special issue on Memory Based Learning.* Journal for Experimental and Theoretical Artificial Intelligence.

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

A. Van den Bosch and W. Daelemans. 1999. Memory-based morphological analysis. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 285–292.

Peter Dienes and Amit Dubey. 2003a. Antecedent recovery: Experiments with a trace tagger. In *Proceedings of the Conference on Empirical Methods in NLP*, pages 33–40.

Peter Dienes and Amit Dubey. 2003b. Deep syntactic processing by combining shallow methods. In *Proceedings of the 41st annual meeting of the Association for Compuatational Linguistics*, pages 431–438.

Thomas G. Dietterich. 1998. Approximate statistical test for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1923.

Saso Dzeroski and Bernard Zenko. 2004. Forgetting exceptions is harmful in language learning. *Machine Learning*, 54(3):255 – 273.

B. S. Everitt. 1977. *The analysis of contingency tables*. Chapman and Hall, London.

Raquel Fernández, Jonathan Ginzburg, and Shalom Lappin. 2004. Classifying ellipsis in dialogue: A machine learning approach. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 240–246.

Raquel Fernández, Jonathan Ginzburg, and Shalom Lappin. 2005. Automatic bare sluice disambiguation in dialogue. In *Proceedings of the 6th International Workshop on Computational Semantics (IWCS-6)*, pages 115–127, Tilburg, The Netherlands.

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

E. Fix and J. Hodges. 1952. Discriminatory analysis: Small sample performance. Technical Report 21-49-004, USAF School of Aviation Medicine, Randolph Field, TX.

Y. Freund and R.E. Schapire. 1999. A short introduction to boosting. 14(5):771–780.

John Goldsmith. 2001. Unsupervised learning of the morphology of a natural language. 27(2):153 – 198.

I. J. Good. 1953. The population frequencies of species and the estimation of population parameters. 40(16):237–264.

Howard Gregory and Shalom Lappin. 1997. A computational model of ellipsis resolution. In *Formal Grammar Conference*, Aix-en-Provence.

Isabelle Haik. 1987. Bound variables that need to be. *Linguistics and Philosophy*, (10):503–530.

Jorge Hankamer and Ivan Sag. 1976. Deep and surface anaphora. *Linguistics Inquiry*, (7):391–426.

Jorge Hankamer. 1979. *Deletion in coordinate structures*. Ph.D. thesis, Yale University.

Daniel Hardt. 1992a. An algorithm for vp ellipsis. In *Proceedings, 29th Annual Meeting of the Association for Computational Linguistics*, Newark, DE.

Daniel Hardt. 1992b. Vp ellipsis and contextual interpretation. In *Proceedings of the International Conference on Computational Linguistics (COLING-92)*, Nantes.

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).

Daniel Hardt. 1998. Improving ellipsis resolution with transformation-based learning. In *AAAI Fall Symposium*.

Daniel Hardt. 1999. Dynamic interpretation of verb phrase ellipsis. *Linguistics and Philosophy*, 2(22):187–221.

Daniel Hardt. 2001. Transformation-based learning of danish grammar correction. In *Proceedings of RANLP*.

Masahiko Haruno, Satoshi Shirai, and Yoshifumi Ooyama. 1998. Using decision trees to construct a practical parser. In *Proceedings of the 17th international conference on Computational linguistics*, pages 505 – 511.

S. Haykin. 1994. *Neural Networks, a Comprehensive Foundation*. Macmillan, New York, NY.

Arild Hestvik. 1995. Reflexives and ellipsis. *Natural Language Semantics*, 3:211–237.

Derrick Higgins and Jerrold M. Sadock. 2003. A machine learning approach to modelling scope preferences. *Computational Linguistics*, 29:73–96.

Veronique Hoste, Walter Daelemans, Iris Hendrickx, and Antal van den Bosch. 2002. Evaluating the results of a memory-based word-expert approach to unrestricted word sense disambiguation. In *Proceedings of the Workshop on word sense disambiguation: Recent successes and future directions*, pages 95–101.

E.T. Jaynes. 1957a. Information theory and statistical mechanics. *Phys. Rev.*, 106:620–630.

E.T. Jaynes. 1957b. Information theory and statistical mechanics. ii. *Phys. Rev.*, 108:171–190.

E.T. Jaynes. 1965. Gibbs vs. boltzmann entropies. *Am. J. Phys.*, 33:391–398.

E.T. Jaynes. 1967. Foundations of probability theory and statistical mechanics. In M.Bunge, editor, *Delaware Seminar in the Foundations of Physics*, pages 77–101. Springer-Verlag, Berlin.

Valentin Jijkoun and Maarten de Rijke. 2004. Enriching the output of a parser using memory-based learning. In *Proceedings of the 42nd annual meeting of the Association for Compuatational Linguistics*, pages 312–319.

Mark Johnson, Stuart Geman, Stephen Canon, Zhiyi Chi, and Stefan Riezler. 1999. Estimators for stochastic 'unification-based' grammars. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 535–541.

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*.

Daniel Jurafsky and James H. Martin. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition*. Prentice-Hall.

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.

Andrew Kehler. 1995. *Interpreting Cohesive Forms in the Context of Discourse Inference*. Ph.D. thesis, Harvard University.

Andrew Kehler. 2002a. Another problem for syntactic (and semantic) theories of ellipsis. *Snippets*, (5):10–11.

Andrew Kehler. 2002b. *Coherence, Reference, and the Theory of Grammar*. CSLI Lecture Notes no. 104, CSLI Publications, Stanford, CA.

Christopher Kennedy and Jason Merchant. 2000. Attributive comparative deletion. *Natural Language and Linguistic Theory*, (18):89–146.

Yoshihisa Kitagawa. 1991. Copying identity. *Natural Language and Linguistic Theory*, (9):497–536.

Dan Klein and Chris Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*.

Sandra Kubler. 2004. *Memory-Based Parsing*. Natural Language Processing 7.

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.

Pat Langley. 1996. *Elements of Machine Learning*. Morgan Kaufmann.

Shalom Lappin and Herbert Leass. 1994. A syntactically based algorithm for pronominal anaphora resolution. *Computational Linguistics*, (20):535–561.

Shalom Lappin and Michael McCord. 1990. Anaphora resolution in slot grammar. *Computational Linguistics*, 16:197–212.

Shalom Lappin and Hsue-Hueh Shih. 1996. A generalized reconstruction algorithm for ellipsis resolution. In *Proceedings of COLING*, pages 687–692.

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.

Shalom Lappin. 2004. A sequenced model of anaphora and ellipsis resolution. In A. Branco, A. McEnery, and R. Mitkov, editors, *Anaphora Processing: Linguistic, Cognitive, and Computational Modelling*. John Benjamins, Amsterdam.

G. Leech, R. Garside, and M. Bryant. 1994. CLAWS-4 : The tagging of the British National Corpus. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING 94)*, pages 622–628, Japan: Kyoto.

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

David D. Lewis and Marc Ringuette. 1994. A comparison of two learning algorithms for text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 81–93, Las Vegas, US.

David M. Magerman. 1994. *Natural Language Parsing as Statistical Pattern Recognition*. Ph.D. thesis, Stanford University.

David M. Magerman. 1995. Statistical decision-tree models for parsing. In *Meeting of the Association for Computational Linguistics*, pages 276–283.

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.

I. Mani and E. Bloedorn. 1998. Machine learning of generic and user-focused summarization. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI'98)*, pages 821–826.

Chris Manning and Hinrich Schutze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA.

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

Mitchell P. Marcus, Bernice Santorini, and Mary Ann Marcinkiewicz. 1994b. Building a large annotated corpus of english : The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Joseph F. McCarthy and Wendy G. Lehnert. 1995. Using decision trees for coreference resolution. In *IJCAI*, pages 1050–1055.

R. Meir and G. Ratsch. 2003. An introduction to boosting and leveraging. In S. Mendelson and A. Smola, editors, *Advanced Lectures on Machine Learning*, pages 119–184. Springer.

S. Meknavin, P. Charoenpornsawat, and B. Kijsirikul. 1997. Feature-based thai word segmentation. In *Proceedings of NLPRS*.

Bernard Merialdo. 1994. Tagging english text with a probabilistic model. *Computational Linguistics*, 20(2):155 – 171.

George A. Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39 – 41.

T. Mitchell. 1997. *Machine Learning*. McGraw-Hill.

Ruslan Mitkov, Richard Evans, and Constantin Orăsan. 2002. A new, fully automatic version of mitkov's knowledge-poor pronoun resolution method. In *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2002)*, Mexico City, Mexico, February, 17 – 23.

G. Ngai and R. Florian. 2001. Transformation-based learning in the fast lane. In *Proceedings of North American ACL*, pages 40–47.

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

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

Leif Arda Nielsen. 2004a. Robust VPE detection using automatically parsed text. In *Proceedings of the Student Workshop, ACL 2004*, pages 31–36, Barcelona, Spain, July.

Leif Arda Nielsen. 2004b. Using automatically parsed text for robust verb phrase ellipsis detection. In *Proceedings of the Fifth Discourse Anaphor and Anaphora Reoslution conference (DAARC)*, pages 121–126, Sao Miguel, Portugal, September.

Leif Arda Nielsen. 2004c. Verb phrase ellipsis detection using automatically parsed text. In *Proceedings of COLING*, pages 1093–1099, Geneva, August.

Leif Arda Nielsen. 2004d. Verb phrase ellipsis detection using machine learning techniques. In N.Nicolov et al., editor, *Recent Advances in Natural Language Processing - vol III (CILT vol 260)*, pages 317–326. Amsterdam & Philadelphia: John Benjamins.

G. Orphanos, D. Kalles, A. Papagelis, and D. Christodoulakis. 1999. Decision trees and nlp: A case study in pos tagging. In *Proceedings of ACAI'99*.

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

Long Qiu, Min-Yen Kan, and Tat-Seng Chua. 2004. A public reference implementation of the rap anaphora resolution algorithm. In *proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC)*, pages 291–294.

R. Quinlan. 1990. Induction of decision trees. In Jude W. Shavlik and Thomas G. Dietterich, editors, *Readings in Machine Learning*. Morgan Kaufmann. Originally published in *Machine Learning* 1:81–106, 1986.

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

Owen Rambow and Daniel Hardt. 2001. Generation of vp ellipsis: a corpus-based approach. In *Proceedings of ACL*.

Lance Ramshaw and Mitchell Marcus. 1995. Text chunking using transformation-based learning. In *Proceedings of the ACL Third Workshop on Very Large Corpora*, pages 82–94.

Adwait Ratnaparkhi. 1996. A maximum entropy part-of-speech tagger. In *Proceedings of the Empirical Methods in Natural Language Processing Conference*.

Adwait Ratnaparkhi. 1997. A linear observed time statistical parser based on maximum entropy models. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*.

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

Adwait Ratnaparkhi. 1998b. Unsupervised statistical models for prepositional phrase attachment. In *Proceedings of the Seventeenth International Conference on Computational Linguistics*.

Adwait Ratnaparkhi. 1999. Learning to parse natural language with maximum entropy models. *Computational Linguistics*, 34:151–175.

Jeffrey C. Reynar and Adwait Ratnaparkhi. 1997. A maximum entropy approach to identifying sentence boundaries. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*.

R. Rosenfeld. 1996. A maximum entropy approach to adaptive statistical language modelling. *Computers, Speech and Language*.

Ivan Sag. 1976. *Deletion and logical form*. Ph.D. thesis, Massachusetts Institute of Technology.

H. Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the International Conference on New Methods in Language Processing*.

Sam Scott and Stan Matwin. 1998. Text classification using WordNet hypernyms. In Sanda Harabagiu, editor, *Use of WordNet in Natural Language Processing Systems: Proceedings of the Conference*, pages 38–44. Association for Computational Linguistics, Somerset, New Jersey.

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

Wee M. Soon, Daniel C. Y. Lim, and Hwee T. Ng. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4).

C. Stanfill and D. Waltz. 1986. Toward memory-based reasoning. *Communications of the ACM*, pages 1213–1228.

Christopher Tancredi. 1992. *Deletion, deaccenting and presupposition.* Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA.

K. M. Ting and I. H. Witten. 1999. Issues in stacked generalization. *Journal of Artificial Intelligence Research*, 10:271–289.

Kristina Toutanova and Christopher Manning. 2002. Feature selection for a rich hpsg grammar using decision trees. In *Proceedings of the Sixth Conference on Natural Language Learning(CoNLL-2002).*

Jorn Veenstra. 1998. Fast np chunking using memory-based learning techniques. In *Proceedings of Benelearn 1998*, pages 71–79.

Jorn Veenstra. 1999. Memory-based text chunking. In *Proceedings of ACAI.*

Gregory Ward and Andrew Kehler. 2002. Syntactic form and discourse accessibility. In *Proceedings of the 4th Discourse Anaphora and Anaphor Resolution Colloquium (DAARC 2002)*, Estoril, Portugal.

Thomas Wasow. 1972. *Anaphoric relations in English.* Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA.

Bonnie Webber, editor. 1979. *A formal approach to discourse anaphora.* New York: Garland Publishing.

Edwin Williams. 1977. Discourse and logical form. *Linguistic Inquiry*, 8(1):101–139.

G. Winter, J.Periaux, and M.Galan, editors. 1995. *Genetic Algorithms in Engineering and Computer Science.* John Wiley and Son Ltd.

D. Wolpert. 1992. Stacked generalization. *Neural Networks*, 2(2):241–260.

Kazuhide Yamamoto and Eiichiro Sumita. 1999. Multiple decision-tree strategy for error-tolerant ellipsis resolution. In *Proceedings of Natural Language Processing Pacific-Rim Symposium (NLPRS'99)*, pages 292–297.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Meeting of the Association for Computational Linguistics*, pages 189–196.

Jakub Zavrel and Walter Daelemans. 1999. Recent advances in memory-based part-of-speech tagging. In *Actas del VI Simposio Internacional de Comunicacion Social*, pages 590–597.

George Kingsley Zipf. 1949. *Human behaviour and the principle of least effort.* Hafner, New York.

Appendix A

Summary of BNC tags

The following descriptions for the tags used in the BNC corpus are taken from the documentation included with the corpus. There are a total of 57 word class tags in the BNC, plus 4 punctuation tags. In addition, there are 30 "Ambiguity Tags", such as AJ0-AV0, which indicates that the choice between adjective (AJ0) and adverb (AV0) is left open, although the tagger has a preference for an adjective reading.

Tag	Description
AJ0	Adjective (general or positive) (e.g. *good, old, beautiful*)
AJC	Comparative adjective (e.g. *better, older*)
AJS	Superlative adjective (e.g. *best, oldest*)
AT0	Article (e.g. *the, a, an, no*)
AV0	General adverb: an adverb not subclassified as AVP or AVQ (see below) (e.g. *often, well, longer* (adv.), *furthest*)
AVP	Adverb particle (e.g. *up, off, out*)
AVQ	Wh-adverb (e.g. *when, where, how, why, wherever*)

continued on next page

continued from previous page

Tag	Description
CJC	Coordinating conjunction (e.g. *and, or, but*)
CJS	Subordinating conjunction (e.g. *although, when*)
CJT	The subordinating conjunction *that*
CRD	Cardinal number (e.g. *one, 3, fifty-five, 3609*)
DPS	Possessive determiner-pronoun (e.g. *your, their, his*)
DT0	General determiner-pronoun: i.e. a determiner-pronoun which is not a DTQ or an AT0.
DTQ	Wh-determiner-pronoun (e.g. *which, what, whose, whichever*)
EX0	Existential there, i.e. *there* occurring in the *there is ...* or *there are ...* construction
ITJ	Interjection or other isolate (e.g. *oh, yes, mhm, wow*)
NN0	Common noun, neutral for number (e.g. *aircraft, data, committee*)
NN1	Singular common noun (e.g. *pencil, goose, time, revelation*)
NN2	Plural common noun (e.g. *pencils, geese, times, revelations*)
NP0	Proper noun (e.g. *London, Michael, Mars, IBM*)
ORD	Ordinal numeral (e.g. *first, sixth, 77th, last*) .
PNI	Indefinite pronoun (e.g. *none, everything, one [as pronoun], nobody*)
PNP	Personal pronoun (e.g. *I, you, them, ours*)
PNQ	Wh-pronoun (e.g. *who, whoever, whom*)
PNX	Reflexive pronoun (e.g. *myself, yourself, itself, ourselves*)
POS	The possessive or genitive marker *'s* or *'*
PRF	The preposition *of*
PRP	Preposition (except for *of*) (e.g. *about, at, in, on, on behalf of, with*)

continued on next page

continued from previous page

Tag	Description
PUL	Punctuation: left bracket - i.e. *(* or *[*
PUN	Punctuation: general separating mark - i.e. *. , ! , : ;* - or *?*
PUQ	Punctuation: quotation mark - i.e. *'* or *"*
PUR	Punctuation: right bracket - i.e. *)* or *]*
TO0	Infinitive marker *to*
UNC	Unclassified items which are not appropriately considered as items of the English lexicon.
VBB	The present tense forms of the verb BE, except for *is, 's*: i.e. *am, are, 'm, 're* and *be* [subjunctive or imperative]
VBD	The past tense forms of the verb BE: *was* and *were*
VBG	The -ing form of the verb BE: *being*
VBI	The infinitive form of the verb BE: *be*
VBN	The past participle form of the verb BE: *been*
VBZ	The -s form of the verb BE: *is, 's*
VDB	The finite base form of the verb BE: *do*
VDD	The past tense form of the verb DO: *did*
VDG	The -ing form of the verb DO: *doing*
VDI	The infinitive form of the verb DO: *do*
VDN	The past participle form of the verb DO: *done*
VDZ	The -s form of the verb DO: *does, 's*
VHB	The finite base form of the verb HAVE: *have, 've*
VHD	The past tense form of the verb HAVE: *had, 'd*
VHG	The -ing form of the verb HAVE: *having*
VHI	The infinitive form of the verb HAVE: *have*
VHN	The past participle form of the verb HAVE: *had*
VHZ	The -s form of the verb HAVE: *has, 's*

continued on next page

	continued from previous page
Tag	Description
VM0	Modal auxiliary verb (e.g. *will, would, can, could, 'll, 'd*)
VVB	The finite base form of lexical verbs (e.g. *forget, send, live, return*) [Including the imperative and present subjunctive]
VVD	The past tense form of lexical verbs (e.g. *forgot, sent, lived, returned*)
VVG	The -ing form of lexical verbs (e.g. *forgetting, sending, living, returning*)
VVI	The infinitive form of lexical verbs (e.g. *forget, send, live, return*)
VVN	The past participle form of lexical verbs (e.g. *forgotten, sent, lived, returned*)
VVZ	The -s form of lexical verbs (e.g. *forgets, sends, lives, returns*)
XX0	The negative particle *not* or *n't*
ZZ0	Alphabetical symbols (e.g. *A, a, B, b, c, d*)

Table A.1: Summary of BNC tags

Appendix B

Summary of Treebank tags

Tag	Description
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential *there*
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural

continued on next page

continued from previous page

Tag	Description
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun
PRP$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	*to*
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun
WP$	Possessive wh-pronoun
WRB	Wh-adverb

Table B.1: Summary of Treebank tags

Appendix C

Summary of RASP tags

CLAWS2 TAGLIST

(The second column of tags, containing an asterisk, are known as 'cover tags', and are used in the development of automatic parsing techniques. They are not relevant to the task of post-editing. They are include for interest because they do occur in some types of CLAWS2 output)

```
!    .*  punctuation tag - exclamation mark
"    "   punctuation tag - quotation marks
$    $*  germanic genitive marker - (' or 's)
&FO  N*  formula
&FW  N*  foreign word
(    (   punctuation tag - left bracket
)    )   punctuation tag - right bracket
,    ,*  punctuation tag - comma
-    ,*  punctuation tag - dash
-----    ^   new sentence marker
.    .*  punctuation tag - full-stop
...  .*  punctuation tag - ellipsis
```

```
:    ;*  punctuation tag - colon
;    ;*  punctuation tag - semi-colon
?    ?*  punctuation tag - question-mark
APP$     A*  possessive pronoun, pre-nominal (my, your, our etc.)
AT   A*  article (the, no)
AT1  A*  singular article (a, an, every)
BCS  EX* before-conjunction (in order (that), even (if etc.))
BTO  EX* before-infinitive marker (in order, so as (to))
CC       CC* coordinating conjunction (and, or)
CCB      CC* coordinating conjunction (but)
CF       CC* semi-coordinating conjunction (so, then, yet)
CS       CS*     subordinating conjunction (if, because, unless)
CSA      CS*     'as' as a conjunction
CSN      CS*     'than' as a conjunction
CST  CS*     'that' as a conjunction
CSW  CS* 'whether' as a conjunction
DA   D*  after-determiner (capable of pronominal function)
             (such, former, same)
DA1  D*  singular after-determiner (little, much)
DA2  D*  plural after-determiner (few, several, many)
DA2R     D*R comparative plural after-determiner (fewer)
DAR  D*R comparative after-determiner (more, less)
DAT  D*      superlative after-determiner (most, least)
DB   D*  before-determiner (capable of pronominal function)
             (all, half)
DB2  D*  plural before-determiner (capable of pronominal
             function) (eg. both)
DD   D*  determiner  (capable of pronominal function) (any, some)
DD1      D*      singular determiner (this, that, another)
DD2  D*  plural determiner (these, those)
DDQ  D*Q wh-determiner (which, what)
DDQ$     D*Q wh-determiner, genitive (whose)
DDQV     D*Q wh-ever determiner (whichever, whatever)
EX   EX* existential 'there'
```

```
ICS ICS*    preposition-conjunction (after, before, since, until)
IF  IF* 'for' as a preposition
II  I*  preposition
IO  IO* 'of' as a preposition
IW  I*  'with'; 'without' as preposition
JA  J*  predicative adjective (tantamount, afraid, asleep)
JB  J*  attributive adjective (main, chief, utter)
JBR J*R attributive comparative adjective (upper, outer)
JBT J*  attributive superlative adjective (utmost, uttermost)
JJ  J*  general adjective
JJR J*R general comparative adjective (older, better, bigger)
JJT J*  general superlative adjective (oldest, best, biggest)
JK  J*  adjective catenative ('able' in 'be able to';
            'willing' in 'be willing to')
LE      UH* leading co-ordinator ('both' in 'both...and...';
            'either' in 'either... or...')
MC  M*  cardinal number neutral for number (two, three...)
MC$ M*  genitive cardinal number, neutral for number (10's)
MC-MC   M*  hyphenated number 40-50, 1770-1827)
MC1 M*  singular cardinal number (one)
MC2 M*      plural cardinal number (tens, twenties)
MD  MD* ordinal number (first, 2nd, next, last)
MF  M*  fraction, neutral for number (quarters, two-thirds)
NC2 N*  plural cited word ('ifs' in 'two ifs and a but')
ND1 N*  singular noun of direction (north, southeast)
NN  N*  common noun, neutral for number (sheep, cod)
NN1 N*  singular common noun (book, girl)
NN1$    N*  genitive singular common noun (domini)
NN2     N*  plural common noun (books, girls)
NNJ N*  organization noun, neutral for number (department,
            council, committee)
NNJ1    N*  singular organization noun (Assembly, commonwealth)
NNJ2    N*  plural organization noun (governments, committees)
NNL N*  locative noun, neutral for number (Is.)
```

```
NNL1    N*  singular locative noun (street, Bay)
NNL2    N*  plural locative noun (islands, roads)
NNO M*  numeral noun, neutral for number (dozen, thousand)
NNO1    M*  singular numeral noun (no known examples)
NNO2    M*  plural numeral noun (hundreds, thousands)
NNS N*  noun of style, neutral for number (no known examples)
NNS1    N*  singular noun of style (president, rabbi)
NNS2    N*  plural noun of style (presidents, viscounts)
NNSA1   N*  following noun of style or title, abbreviatory (M.A.)
NNSA2   N*  following plural noun of style or title, abbreviatory
NNSB    N*  preceding noun of style or title, abbr. (Rt. Hon.)
NNSB1   N*  preceding sing. noun of style or title, abbr. (Prof.)
NNSB2   N*  preceding plur. noun of style or title, abbr. (Messrs.)
NNT N*  temporal noun, neutral for number (no known examples)
NNT1    N*  singular temporal noun (day, week, year)
NNT2    N*  plural temporal noun (days, weeks, years)
NNU N*  unit of measurement, neutral for number (in., cc.)
NNU1    N*  singular unit of measurement (inch, centimetre)
NNU2    N*  plural unit of measurement (inches, centimetres)
NP  N*  proper noun, neutral for number (Indies, Andes)
NP1 N*  singular proper noun (London, Jane, Frederick)
NP2 N*  plural proper noun (Browns, Reagans, Koreas)
NPD1    N*  singular weekday noun (Sunday)
NPD2    N*  plural weekday noun (Sundays)
NPM1    N*  singular month noun (October)
NPM2    N*  plural month noun (Octobers)
PN  P*  indefinite pronoun, neutral for number ("none")
PN1 P*  singular indefinite pronoun (one, everything, nobody)
PNQO    P*Q whom
PNQS    P*Q who
PNQV$   P*Q whosever
PNQVO   P*Q whomever, whomsoever
PNQVS   P*Q whoever, whosoever
PNX1    P*  reflexive indefinite pronoun (oneself)
```

```
PP$ P*  nominal possessive personal pronoun (mine, yours)
PPH1    P*  it
PPHO1   P*  him, her
PPHO2   P*  them
PPHS1   P*S he, she
PPHS2   P*S they
PPIO1   P*  me
PPIO2   P*  us
PPIS1   P*S I
PPIS2   P*S we
PPX1    P*  singular reflexive personal pronoun (yourself, itself)
PPX2    P*  plural reflexive personal pronoun (yourselves,
            ourselves)
PPY P*  you
RA  R*  adverb, after nominal head (else, galore)
REX R*  adverb introducing appositional constructions
            (namely, viz, eg.)
RG  RG* degree adverb (very, so, too)
RGA R*  post-nominal/adverbial/adjectival degree adverb
            (indeed, enough)
RGQ RGQ*    wh- degree adverb (how)
RGQV    RGQ*    wh-ever degree adverb (however)
RGR RGR*    comparative degree adverb (more, less)
RGT RG* superlative degree adverb (most, least)
RL  R*  locative adverb (alongside, forward)
RP  RP* prep. adverb; particle (in, up, about)
RPK RP* prep. adv., catenative ('about' in 'be about to')
RR  R*  general adverb
RRQ R*Q wh- general adverb (where, when, why, how)
RRQV    R*Q wh-ever general adverb (wherever, whenever)
RRR R*R comparative general adverb (better, longer)
RRT R*  superlative general adverb (best, longest)
RT  NR* nominal adverb of time (now, tommorow)
TO  TO* infinitive marker (to)
```

```
UH  UH* interjection (oh, yes, um)
VB0 VB0*    be
VBDR    VB0*    were
VBDZ    VB0*    was
VBG VBG*    being
VBM VB0*    am
VBN VBN*    been
VBR VB0*    are
VBZ VB0*    is
VD0 VD0*    do
VDD VD0*    did
VDG VDG*    doing
VDN     VDN*    done
VDZ VD0*    does
VH0 VH0*    have
VHD VH0*    had (past tense)
VHG VHG*    having
VHN VHN*    had (past participle)
VHZ VH0*    has
VM  VD0*    modal auxiliary (can, will, would etc.)
VMK VD0*    modal catenative (ought, used)
VV0 VV0*    base form of lexical verb (give, work etc.)
VVD VV0*    past tense form of lexical verb (gave, worked etc.)
VVG VVG*    -ing form of lexical verb (giving, working etc.)
VVN VVN*    past participle form of lexical verb (given,
            worked etc.)
VVZ VV0*    -s form of lexical verb (gives, works etc.)
VVGK    VVG*    -ing form in a catenative verb ('going' in
            'be going to')
VVNK    VVN*    past part. in a catenative verb ('bound' in
            'be bound to')
XX  XX* not, n't
ZZ1 N*  singular letter of the alphabet:'A', 'a', 'B', etc.
ZZ2 N*  plural letter of the alphabet: 'As', b's, etc.
```

Appendix D

Detailed tables for VPE detection experiments

D.1 Information contributed by features

The statistics computed by TiMBL on the Treebank data is seen below.

Feats	Vals	X-square	Variance	InfoGain	GainRatio
1	(ignored)				
2	12137	16712.715	0.15244099	0.0039238256	0.00043919421
3	50	58.988505	0.00053804938	0.00040964490	9.4731513e-05
4	11192	6889.3556	0.062839590	0.0029077632	0.00032733878
5	49	151.77082	0.0013843408	0.00068396711	0.00015724614
6	9354	8410.9652	0.076718583	0.0034672035	0.00040863022
7	46	141.68281	0.0012923255	0.00062694908	0.00017237752
8	8103	1292.3308	0.011787682	0.0024136764	0.00027172178
9	8	190.41410	0.0017368161	0.0013083288	0.00048445462
10	10315	15418.036	0.14063189	0.0052103150	0.00058360551
11	50	251.49628	0.0022939624	0.0014149285	0.00035095910
12	13194	4267.3391	0.038923501	0.0025613633	0.00027148325
13	49	145.80863	0.0013299582	0.00085309118	0.00019923309
14	13210	8828.4326	0.080526411	0.0026015187	0.00027832239
15	50	115.93471	0.0010574704	0.00068647300	0.00015842766

16	2	43.701963	0.00039861688	0.00017766549	0.00053813059
17	2	2956.1612	0.026963909	0.0021685750	0.033044246
18	386	1954.6430	0.017828803	0.00074711840	0.00021347748
19	146	115.13917	0.0010502141	0.00037763038	0.00015736589
20	2	4673.1456	0.042624967	0.0032076269	0.042644430
21	2	17291.932	0.15772417	0.0041891290	0.17979602

Feature Permutation based on GainRatio/Values :
< 21, 20, 17, 16, 9, 11, 13, 7, 5, 15, 3, 19, 18, 10, 6, 2, 8, 4, 14, 12, 1 >

where the features are[1] :

2	$word_{-3}$
3	tag_{-3}
4	$word_{-2}$
5	tag_{-2}
6	$word_{-1}$
7	tag_{-1}
8	$word$
9	tag
10	$word_{+1}$
11	tag_{+1}
12	$word_{+2}$
13	tag_{+2}
14	$word_{+3}$
15	tag_{+3}
16	$closetopunctuation$
17	$heuristic$
18	$previouscategory$
19	$nextcategory$
20	$Auxiliary - finalVP$
21	$EmptyVP$

[1]Feature 1 is the identifier of the example and is ignored

D.2 Detailed results on re-parsed Treebank data

D.2.1 Using Charniak's parser

	Recall	Precision	F1	|ER|	%ER
Words + POS	45.33	49.27	47.22	-	-
+ group	54.00	62.30	57.85	10.63	20.14
+ close-to-punct	54.00	61.36	57.44	-0.41	-0.97
+ heuristic baseline	56.66	64.39	60.28	2.84	6.67
+ surrounding cat.	56.66	64.39	60.28	0	0
+ auxiliary-final VP	58.00	65.41	61.48	1.2	3.02
+ empty VP	61.33	70.22	65.48	4.00	10.38
+ empty categories	**62.66**	**71.21**	**66.66**	1.18	3.42

Table D.1: Results on data from the Treebank parsed with Charniak's parser - MBL

	Recall	Precision	F1	|ER|	%ER
Words + POS	19.33	76.31	30.85	-	-
+ group	38.66	79.45	52.01	21.16	30.60
+ close-to-punct	40.00	80.00	53.33	1.32	2.75
+ heuristic baseline	41.33	75.60	53.44	0.11	0.24
+ surrounding cat.	42.66	77.10	54.93	1.49	3.20
+ auxiliary-final VP	50.66	73.78	60.07	5.14	11.40
+ empty VP	**53.33**	**74.76**	**62.25**	2.18	5.46
+ empty categories	48.00	70.58	57.14	-5.11	-13.54

Table D.2: Results on data from the Treebank parsed with Charniak's parser - GIS-MaxEnt

	Recall	Precision	F1	|ER|	%ER
Words + POS	49.33	80.43	61.15	-	-
+ group	56.66	71.42	63.19	2.04	5.25
+ close-to-punct	54.00	69.23	60.67	-2.52	-6.85
+ heuristic baseline	62.00	71.53	66.42	5.75	14.62
+ surrounding cat.	52.00	78.78	62.65	-3.77	-11.23
+ auxiliary-final VP	65.33	72.05	68.53	5.88	15.74
+ empty VP	**68.00**	**75.00**	**71.32**	2.79	8.87
+ empty categories	64.66	72.93	68.55	-2.77	-9.66

Table D.3: Results on data from the Treebank parsed with Charniak's parser - L-BFGS-MaxEnt

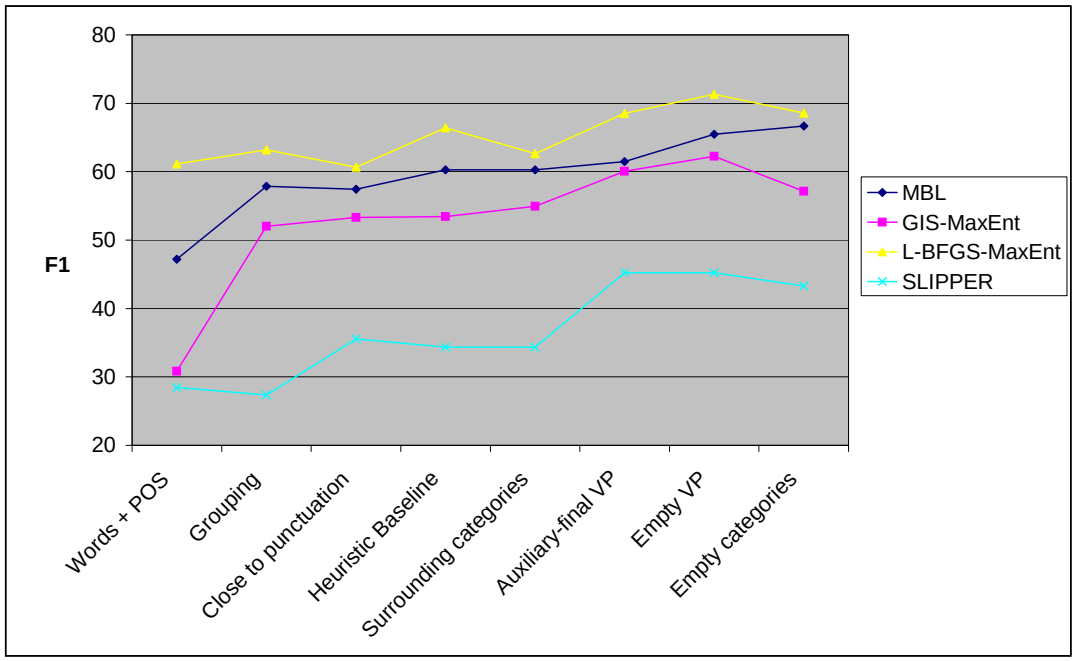

Figure D.1: F1 plot for algorithms on Charniak parsed Treebank data versus features being added

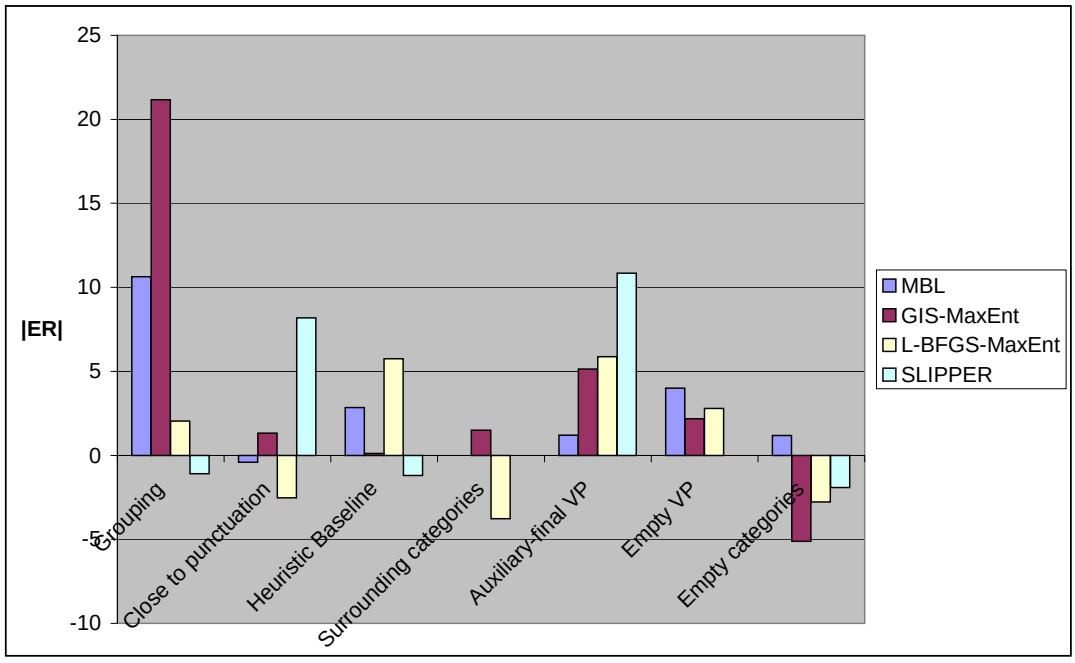

Figure D.2: Error Reduction effect of features on Charniak parsed Treebank data

	Recall	Precision	F1	|ER|	%ER
Words + POS	23.33	36.46	28.46	-	-
+ group	26.00	28.89	27.37	-1.09	-1.52
+ close-to-punct	42.67	30.48	35.56	8.19	11.28
+ heuristic baseline	55.33	24.92	34.37	-1.19	-1.85
+ surrounding cat.	55.33	24.92	34.37	0	0
+ auxiliary-final VP	**66.00**	**34.38**	**45.21**	10.84	16.52
+ empty VP	66.00	34.38	45.21	0	0
+ empty categories	61.33	33.45	43.29	-1.92	-3.50

Table D.4: Results on data from the Treebank parsed with Charniak's parser - SLIPPER

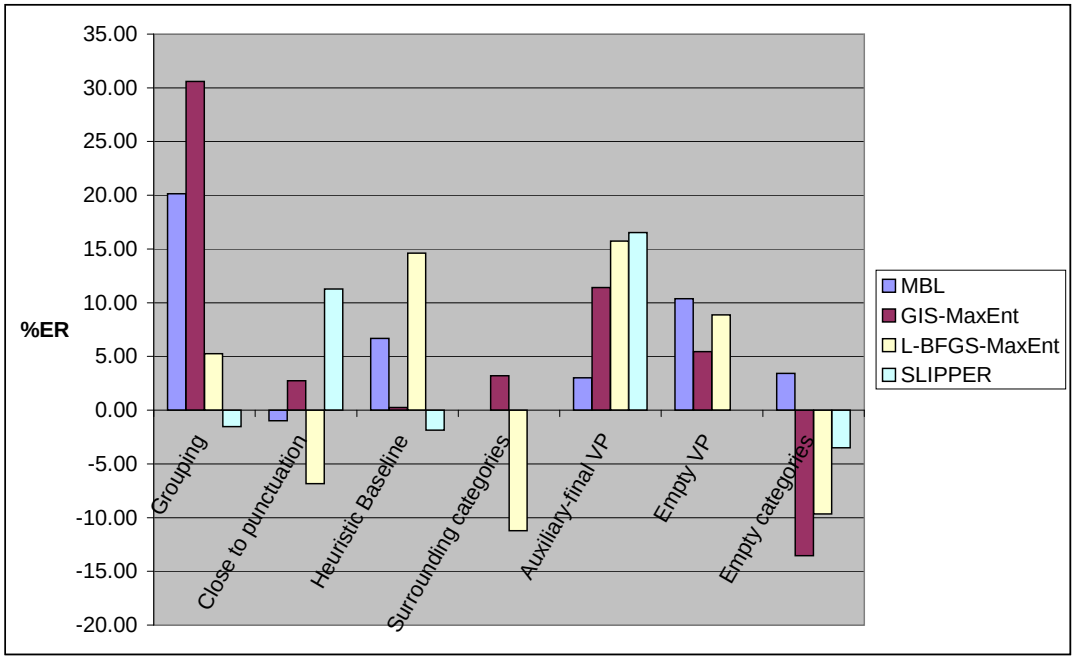

Figure D.3: Percentage Error Reduction effect of features on Charniak parsed Treebank data

D.2.2 Using RASP

	Recall	Precision	F1	|ER|	%ER
Words + POS	50.98	68.42	58.42	-	-
+ group	56.86	67.96	61.92	3.5	8.42
+ lemma	55.55	69.10	61.59	-0.33	-0.87
+ close-to-punct	56.20	69.91	62.31	0.72	1.87
+ heuristic baseline	**59.47**	**74.59**	**66.18**	3.87	10.27
+ surrounding cat.	59.47	72.22	65.23	-0.95	-2.81
+ auxiliary-final VP	59.47	70.54	64.53	-0.7	-2.01

Table D.5: Results on data from the Treebank parsed with RASP - MBL

	Recall	Precision	F1	|ER|	%ER
Words + POS	21.56	56.89	31.27	-	-
+ group	28.10	74.13	40.75	9.48	13.79
+ lemma	30.06	79.31	43.60	2.85	4.81
+ close-to-punct	39.21	80.00	52.63	9.03	16.01
+ heuristic baseline	49.67	74.50	59.60	6.97	14.71
+ surrounding cat.	**56.86**	**75.00**	**64.68**	5.08	12.57
+ auxiliary-final VP	56.86	74.35	64.44	-0.24	-0.68

Table D.6: Results on data from the Treebank parsed with RASP - GIS-MaxEnt

	Recall	Precision	F1	|ER|	%ER
Words + POS	50.32	79.38	61.60	-	-
+ group	51.63	79.00	62.45	0.85	2.21
+ lemma	50.32	79.38	61.60	-0.85	-2.26
+ close-to-punct	47.71	79.34	59.59	-2.01	-5.23
+ heuristic baseline	56.20	73.50	63.70	4.11	10.17
+ surrounding cat.	60.13	77.31	67.64	3.94	10.85
+ auxiliary-final VP	**62.09**	**77.86**	**69.09**	1.45	4.48

Table D.7: Results on data from the Treebank parsed with RASP - L-BFGS-MaxEnt

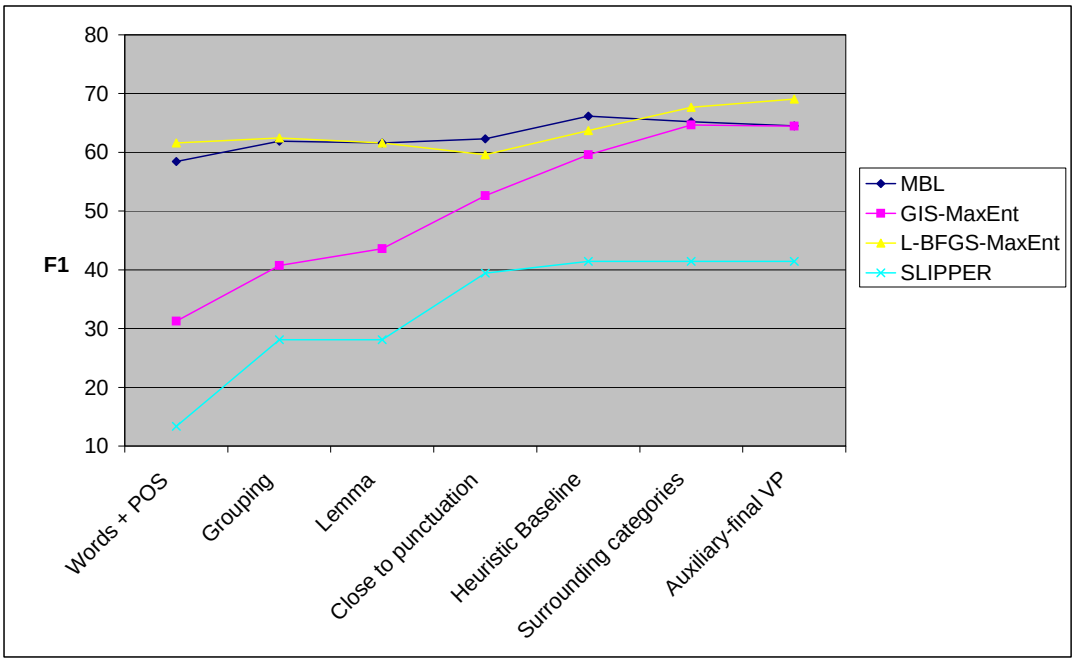

Figure D.4: F1 plot for algorithms on RASP parsed Treebank data versus features being added

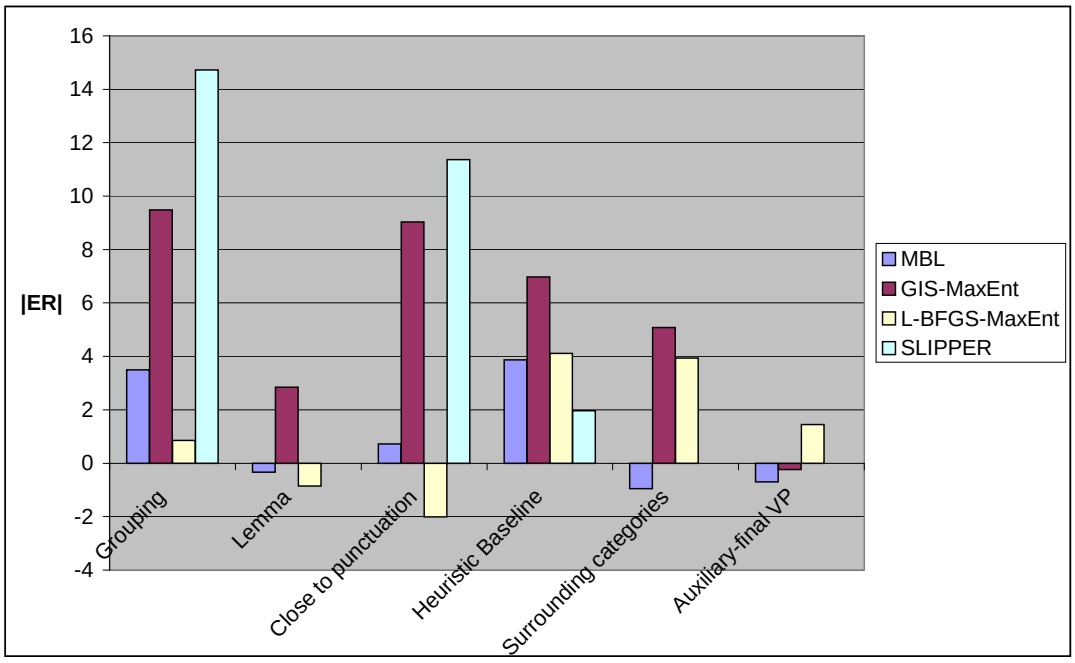

Figure D.5: Error Reduction effect of features on RASP parsed Treebank data

	Recall	Precision	F1	|ER|	%ER
Words + POS	23.53	9.33	13.36	-	-
+ group	26.80	29.50	28.08	14.72	16.99
+ lemma	26.80	29.50	28.08	0	0
+ close-to-punct	37.25	41.91	39.45	11.37	15.81
+ heuristic baseline	76.47	28.40	41.42	1.97	3.25
+ surrounding cat.	76.47	28.40	41.42	0	0
+ auxiliary-final VP	76.47	28.40	41.42	0	0

Table D.8: Results on data from the Treebank parsed with RASP - SLIPPER

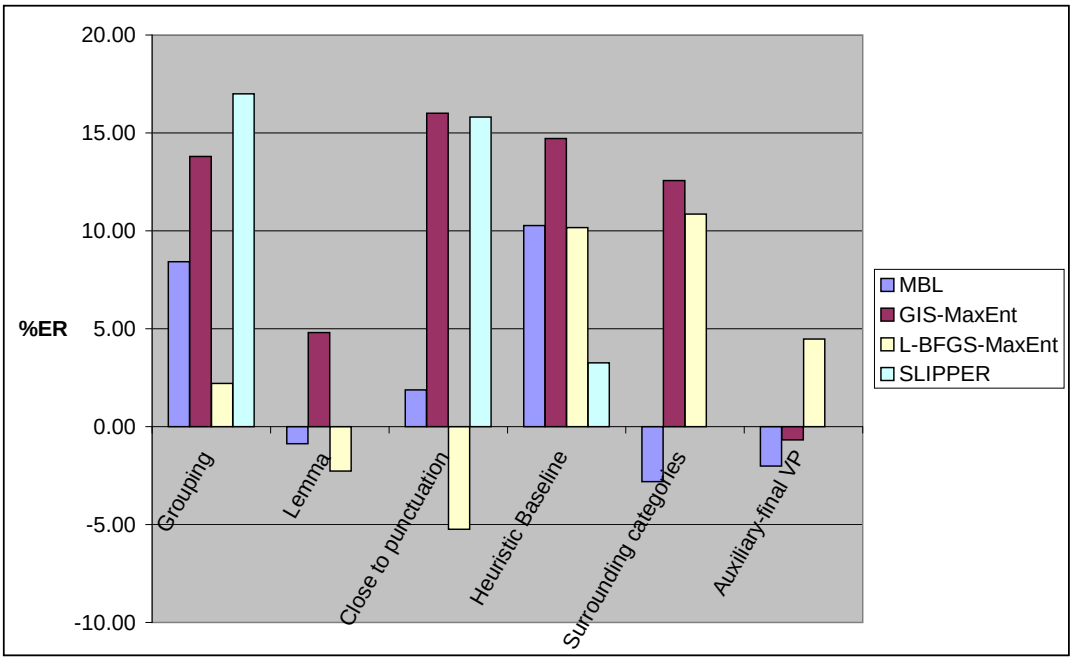

Figure D.6: Percentage Error Reduction effect of features on RASP parsed Treebank data

D.3 Results on re-parsed BNC data

D.3.1 Using Charniak's parser

	Recall	Precision	F1	|ER|	%ER
Words + POS	57.00	58.46	57.72	-	-
+ group	66.5	63.63	65.03	7.31	17.29
+ close-to-punct	65.00	62.50	63.72	-1.31	-3.75
+ heuristic baseline	66.00	61.68	63.76	0.04	0.11
+ surrounding cat.	67.00	65.04	66.00	2.24	6.18
+ auxiliary-final VP	67.50	67.16	67.33	1.33	3.91
+ empty VP	**68.00**	**66.99**	**67.49**	0.16	0.49
+ empty categories	69.00	65.40	67.15	-0.34	-1.05

Table D.9: Results on data from the BNC parsed with Charniak's parser - MBL

	Recall	Precision	F1	|ER|	%ER
Words + POS	39.50	86.81	54.29	-	-
+ group	55.00	75.86	63.76	9.47	20.72
+ close-to-punct	58.49	77.48	66.66	2.9	8.00
+ heuristic baseline	61.50	75.46	67.76	1.1	3.30
+ surrounding cat.	61.00	76.72	67.96	0.2	0.62
+ auxiliary-final VP	**65.00**	**75.58**	**69.89**	1.93	6.02
+ empty VP	65.00	75.14	69.70	-0.19	-0.63
+ empty categories	64.00	72.72	68.08	-1.62	-5.35

Table D.10: Results on data from the BNC parsed with Charniak's parser - GIS-MaxEnt

	Recall	Precision	F1	|ER|	%ER
Words + POS	63.50	72.98	67.91	-	-
+ group	71.00	70.64	70.82	2.91	9.07
+ close-to-punct	70.00	73.68	71.79	0.97	3.32
+ heuristic baseline	**72.50**	**72.86**	**72.68**	0.89	3.15
+ surrounding cat.	71.50	70.09	70.79	-1.89	-6.92
+ auxiliary-final VP	71.00	73.19	72.08	1.29	4.42
+ empty VP	71.50	68.75	70.09	-1.99	-7.13
+ empty categories	74.00	68.83	71.32	1.23	4.11

Table D.11: Results on data from the BNC parsed with Charniak's parser - L-BFGS-MaxEnt

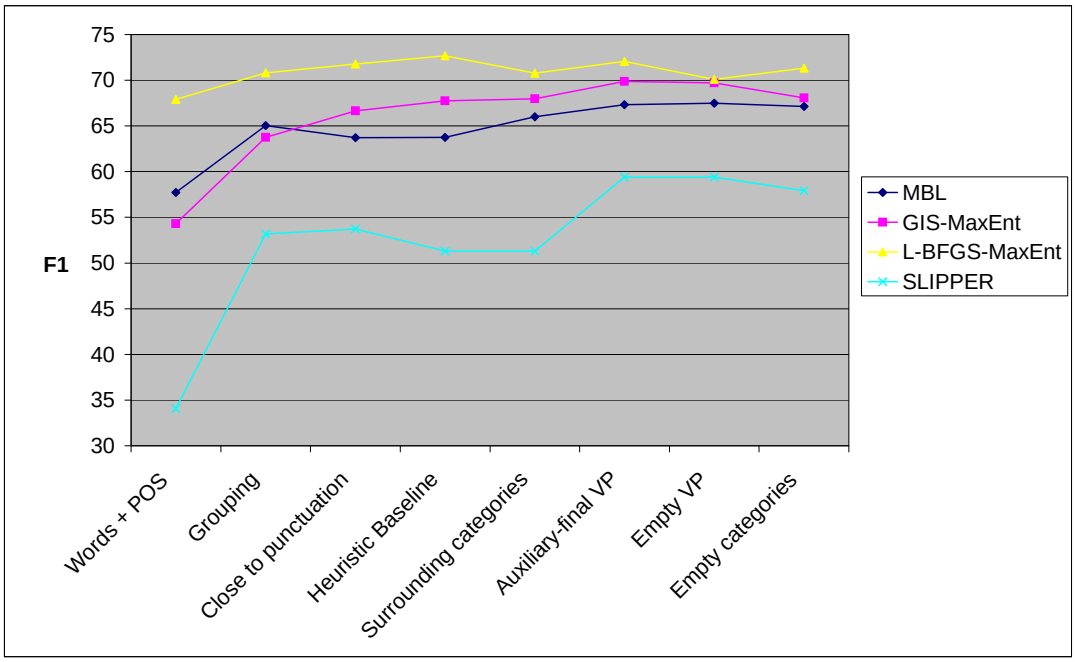

Figure D.7: F1 plot for algorithms on Charniak parsed BNC data versus features being added

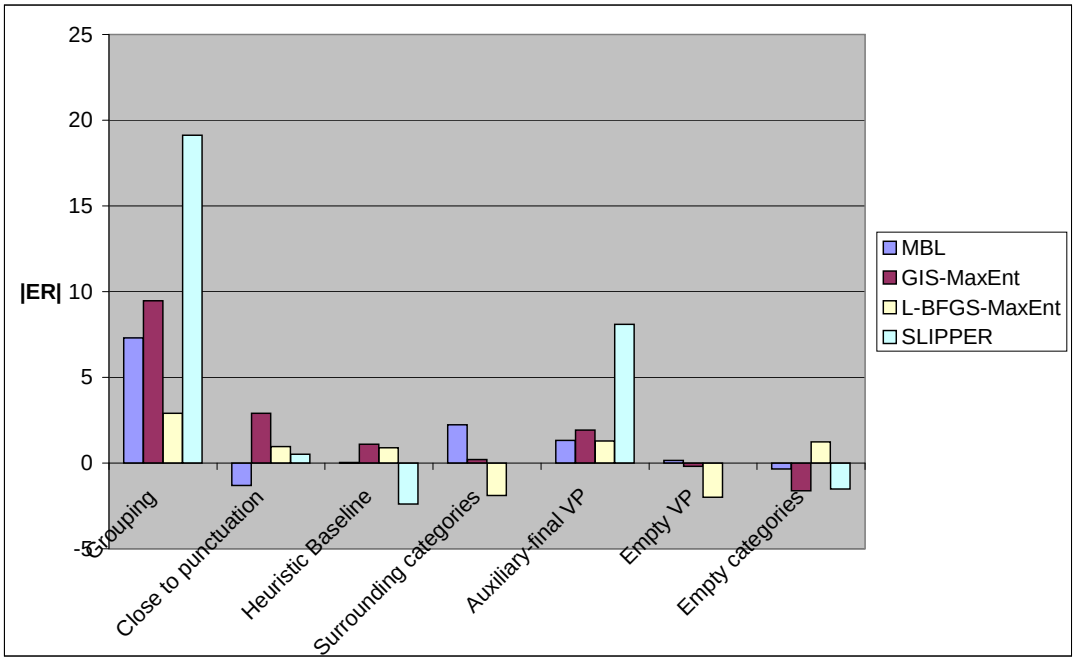

Figure D.8: Error Reduction effect of features on Charniak parsed BNC data

	Recall	Precision	F1	|ER|	%ER
Words + POS	39.00	30.23	34.06	-	-
+ group	48.00	59.63	53.19	19.13	29.01
+ close-to-punct	61.50	47.67	53.71	0.52	1.11
+ heuristic baseline	68.00	41.21	51.32	-2.39	-5.16
+ surrounding cat.	68.00	41.21	51.32	0	0
+ auxiliary-final VP	**75.00**	**49.18**	**59.41**	8.09	16.62
+ empty VP	75.00	49.18	59.41	0	0
+ empty categories	81.50	44.90	57.90	-1.51	-3.72

Table D.12: Results on data from the BNC parsed with Charniak's parser - SLIPPER

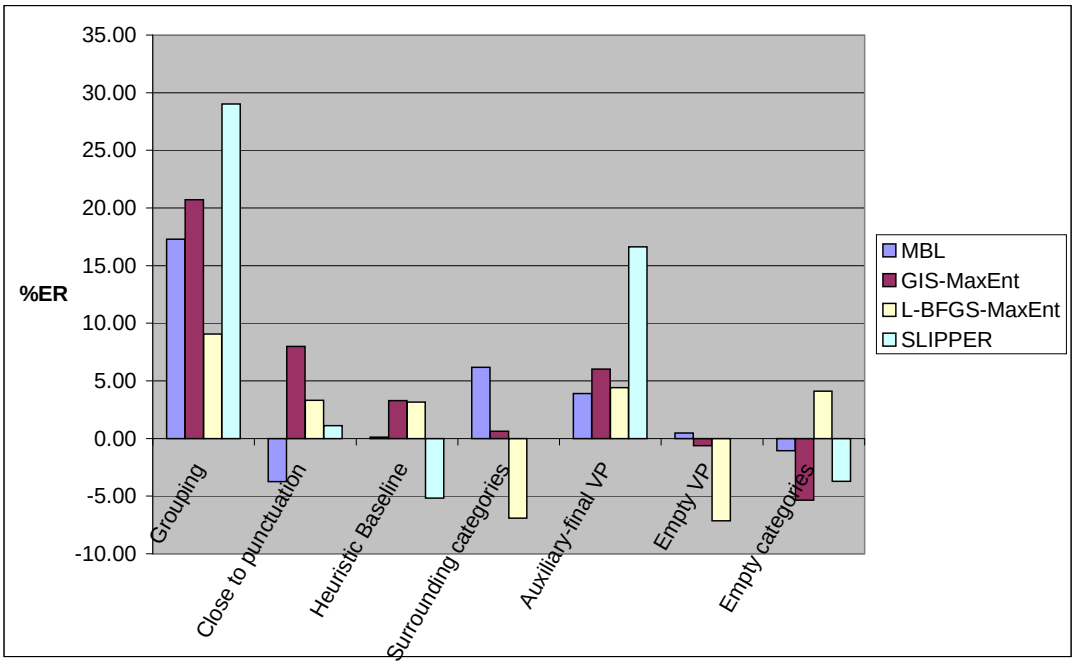

Figure D.9: Percentage Error Reduction effect of features on Charniak parsed BNC data

D.3.2 Using RASP

	Recall	Precision	F1	|ER|	%ER
Words + POS	55.94	65.31	60.26	-	-
+ group	57.42	65.53	61.21	0.95	2.39
+ lemma	59.90	66.85	63.18	1.97	5.08
+ close-to-punct	59.900	70.76	64.87	1.69	4.59
+ heuristic baseline	**61.38**	**76.07**	**67.94**	3.07	8.74
+ surrounding cat.	62.37	73.25	67.37	-0.57	-1.78
+ auxiliary-final VP	62.37	72.00	66.84	-0.53	-1.62

Table D.13: Results on data from the BNC parsed with RASP - MBL

	Recall	Precision	F1	|ER|	%ER
Words + POS	39.60	61.53	48.19	-	-
+ group	45.54	68.14	54.59	6.40	12.35
+ lemma	48.01	71.32	57.39	2.80	6.17
+ close-to-punct	52.47	76.25	62.17	4.78	11.22
+ heuristic baseline	61.38	72.94	66.66	4.49	11.87
+ surrounding cat.	**64.35**	**73.03**	**68.42**	1.76	5.28
+ auxiliary-final VP	64.35	73.03	68.42	0	0

Table D.14: Results on data from the BNC parsed with RASP - GIS-MaxEnt

	Recall	Precision	F1	|ER|	%ER
Words + POS	60.89	72.78	66.30	-	-
+ group	59.40	71.42	64.86	-1.44	-4.27
+lemma	60.39	72.61	65.94	1.08	3.07
+ close-to-punct	59.40	70.58	64.51	-1.43	-4.20
+ heuristic baseline	62.87	72.57	67.37	2.86	8.06
+ surrounding cat.	65.84	72.28	68.91	1.54	4.72
+ auxiliary-final VP	**66.33**	**72.82**	**69.43**	0.52	1.67

Table D.15: Results on data from the BNC parsed with RASP - L-BFGS-MaxEnt

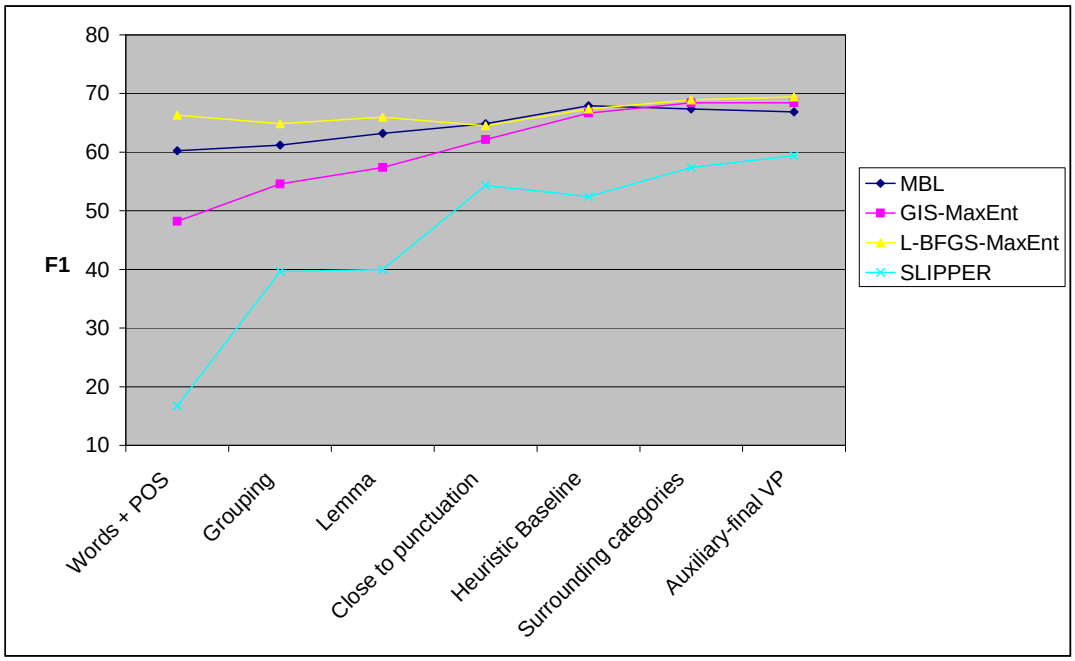

Figure D.10: F1 plot for algorithms on RASP parsed BNC data versus features being added

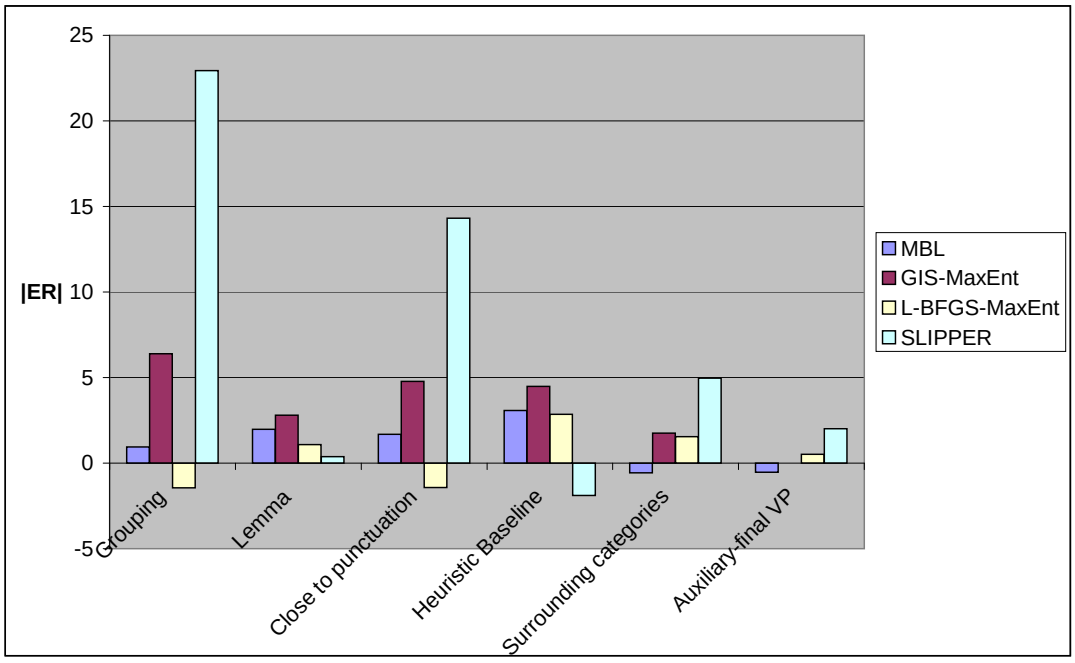

Figure D.11: Error Reduction effect of features on RASP parsed BNC data

	Recall	Precision	F1	|ER|	%ER
Words + POS	21.78	13.54	16.70	-	-
+ group	31.68	52.89	39.63	22.93	27.53
+ lemma	31.68	54.24	40.00	0.37	0.61
+ close-to-punct	54.46	54.19	54.32	14.32	23.87
+ heuristic baseline	80.20	38.94	52.43	-1.89	-4.14
+ surrounding cat.	77.72	45.51	57.40	4.97	10.45
+ auxiliary-final VP	**75.00**	**49.18**	**59.41**	8.09	16.62

Table D.16: Results on data from the BNC parsed with RASP - SLIPPER

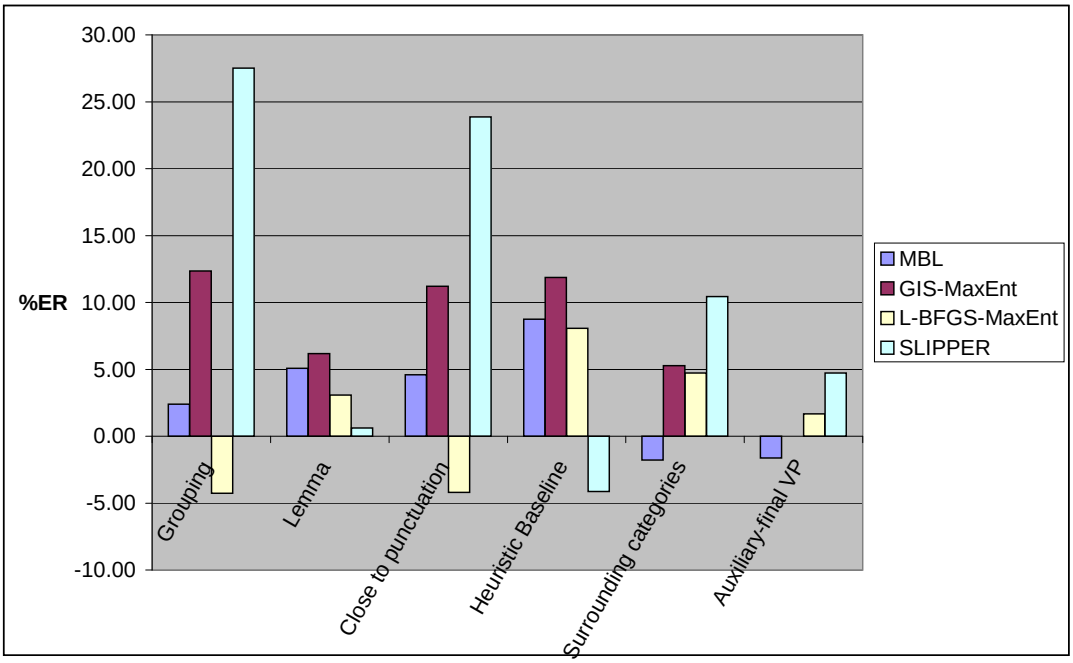

Figure D.12: Percentage Error Reduction effect of features on RASP parsed BNC data

D.4 Results on combined re-parsed data

D.4.1 Using Charniak's parser

	Recall	Precision	F1	|ER|	%ER
Words + POS	55.42	56.72	56.06	-	-
+ group	62.28	69.20	65.56	9.50	21.62
+ close-to-punct	62.57	69.08	65.66	0.10	0.29
+ heuristic baseline	63.14	67.79	65.38	-0.28	-0.82
+ surrounding cat.	62.85	67.48	65.08	-0.30	-0.87
+ auxiliary-final VP	64.28	69.018	66.56	1.48	4.24
+ empty VP	65.71	70.12	67.84	1.28	3.83
+ empty categories	**65.71**	**71.87**	**68.65**	0.81	2.52

Table D.17: Results on combined data parsed with Charniak's parser - MBL

	Recall	Precision	F1	|ER|	%ER
Words + POS	34.85	79.73	48.50	-	-
+ group	54.28	77.86	63.97	15.47	30.04
+ close-to-punct	53.71	76.42	63.08	-0.89	-2.47
+ heuristic baseline	56.85	73.70	64.19	1.11	3.01
+ surrounding cat.	57.42	73.35	64.42	0.23	0.64
+ auxiliary-final VP	**63.71**	**74.83**	**68.82**	4.40	12.37
+ empty VP	63.71	74.08	68.50	-0.32	-1.03
+ empty categories	63.71	72.40	67.78	-0.72	-2.29

Table D.18: Results on combined data parsed with Charniak's parser - GIS-MaxEnt

	Recall	Precision	F1	|ER|	%ER
Words + POS	62.85	75.86	68.75	-	-
+ group	65.14	69.30	67.15	-1.6	-5.12
+ close-to-punct	67.14	67.72	67.43	0.28	0.85
+ heuristic baseline	69.71	69.71	69.71	2.28	7.00
+ surrounding cat.	67.42	70.87	69.10	-0.61	-2.01
+ auxiliary-final VP	**71.71**	**71.30**	**71.50**	2.40	7.77
+ empty VP	60.85	75.80	67.51	-3.99	-14.00
+ empty categories	70.85	69.85	70.35	2.84	8.74

Table D.19: Results on combined data parsed with Charniak's parser - L-BFGS-MaxEnt

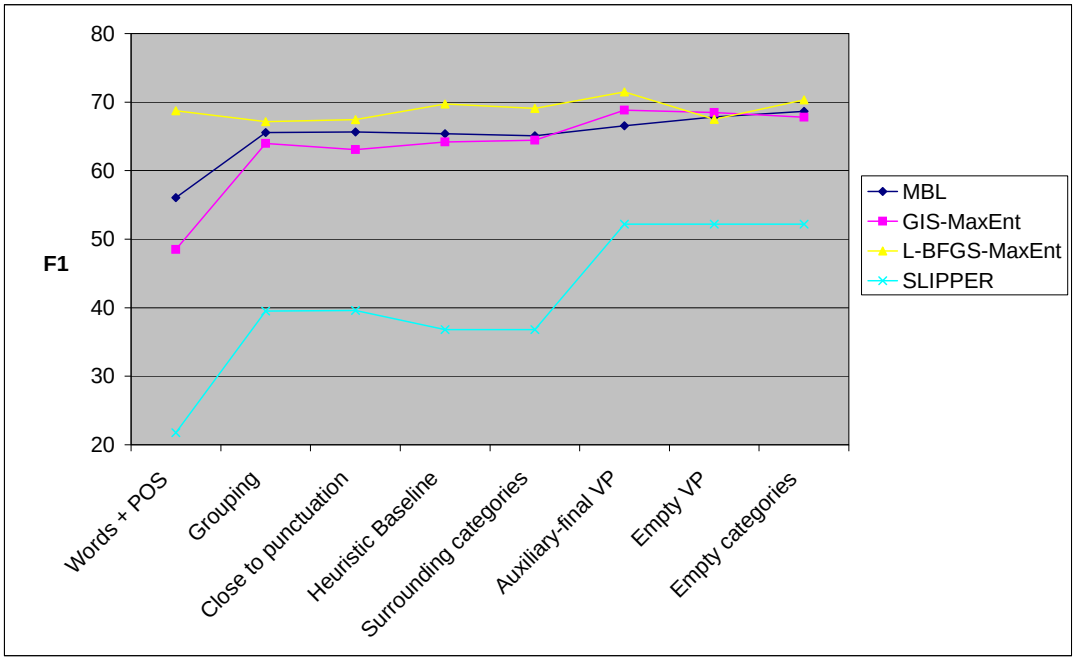

Figure D.13: F1 plot for algorithms on Charniak parsed combined data versus features being added

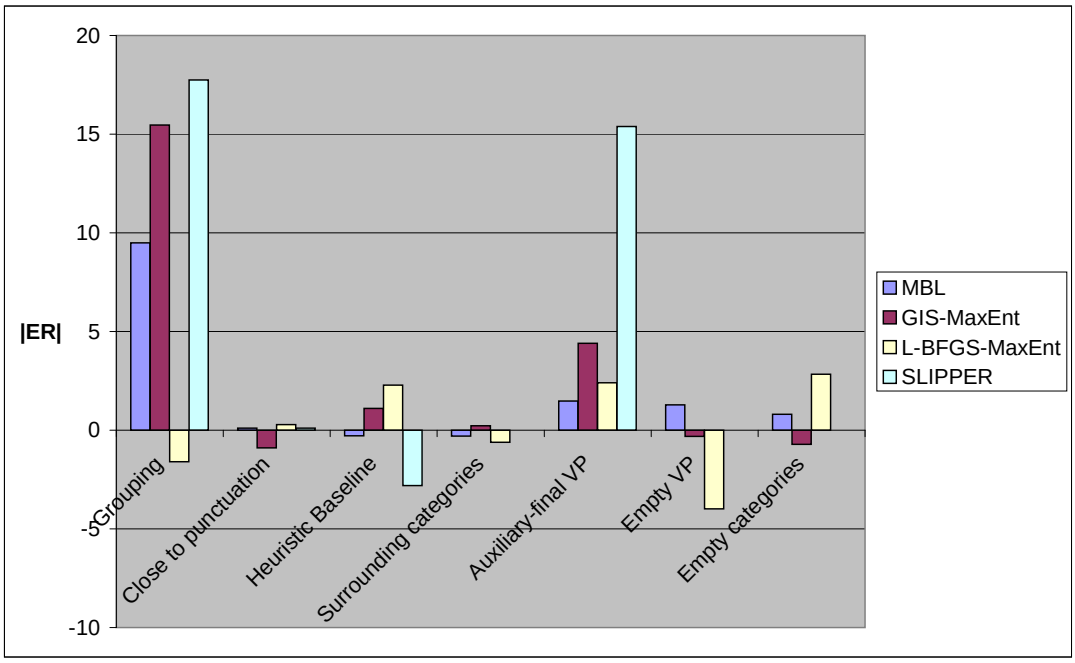

Figure D.14: Error Reduction effect of features on Charniak parsed combined data

	Recall	Precision	F1	|ER|	%ER
Words + POS	32.29	16.42	21.77	-	-
+ group	41.14	37.99	39.51	17.74	22.68
+ close-to-punct	40.29	38.95	39.61	0.10	0.17
+ heuristic baseline	29.71	48.37	36.81	-2.80	-4.64
+ surrounding cat.	29.71	48.37	36.81	0	0
+ auxiliary-final VP	**64.29**	**43.95**	**52.20**	15.39	24.36
+ empty VP	64.29	43.95	52.20	0	0
+ empty categories	64.29	43.95	52.20	0	0

Table D.20: Results on combined data parsed with Charniak's parser - SLIPPER

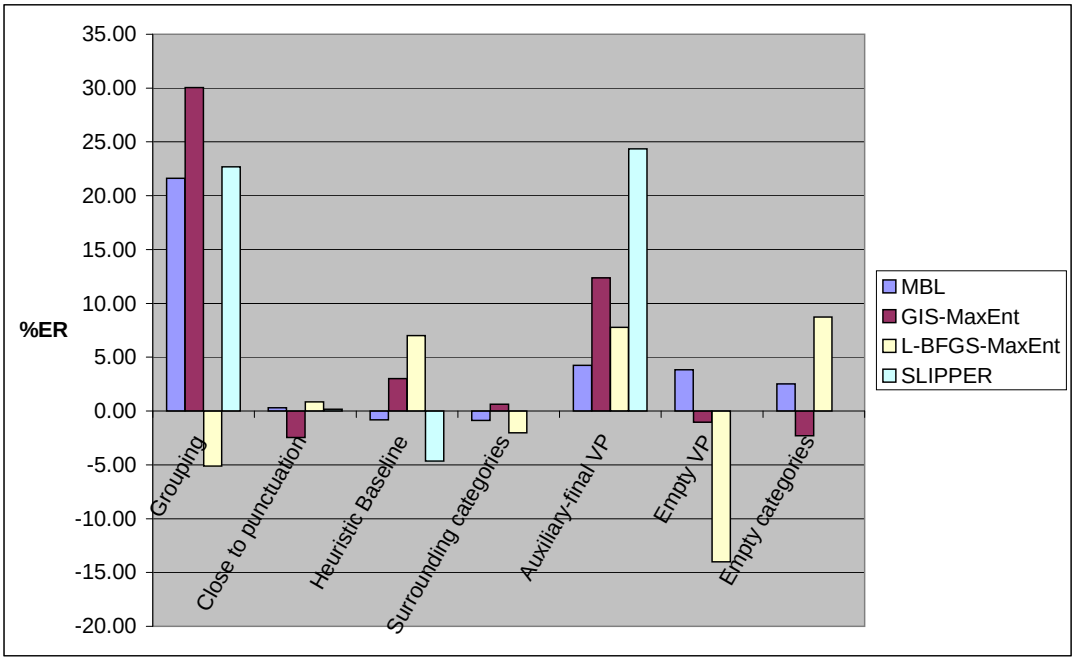

Figure D.15: Percentage Error Reduction effect of features on Charniak parsed combined data

D.4.2 Using RASP

	Recall	Precision	F1	|ER|	%ER
Words + POS	59.15	70.00	64.12	-	-
+ group	61.40	68.76	64.88	0.76	2.12
+ lemma	62.53	69.37	65.77	0.89	2.53
+ close-to-punct	63.09	70.88	66.76	0.99	2.89
+ heuristic baseline	**64.50**	**74.59**	**69.18**	2.42	7.28
+ surrounding cat.	64.22	72.38	68.05	-1.13	-3.67
+ auxiliary-final VP	64.22	70.37	67.15	-0.9	-2.82

Table D.21: Results on combined data parsed with RASP - MBL

	Recall	Precision	F1	|ER|	%ER
Words + POS	36.05	62.74	45.79	-	-
+ group	41.12	71.56	52.23	6.44	11.88
+ lemma	42.53	75.12	54.31	2.08	4.35
+ close-to-punct	46.19	75.22	57.24	2.93	6.41
+ heuristic baseline	61.12	74.31	67.07	9.83	22.99
+ surrounding cat.	61.69	73.98	67.28	0.21	0.64
+ auxiliary-final VP	**61.97**	**73.82**	**67.38**	0.10	0.31

Table D.22: Results on combined data parsed with RASP - GIS-MaxEnt

	Recall	Precision	F1	|ER|	%ER
Words + POS	58.87	71.57	64.60	-	-
+ group	61.69	71.33	66.16	1.56	4.41
+lemma	61.69	71.56	66.26	0.10	0.30
+ close-to-punct	58.87	69.20	63.62	-2.64	-7.82
+ heuristic baseline	62.25	73.42	67.37	3.75	10.31
+ surrounding cat.	66.76	73.60	70.01	2.64	8.09
+ auxiliary-final VP	**70.42**	**71.42**	**70.92**	0.91	3.03

Table D.23: Results on combined data parsed with RASP - L-BFGS-MaxEnt

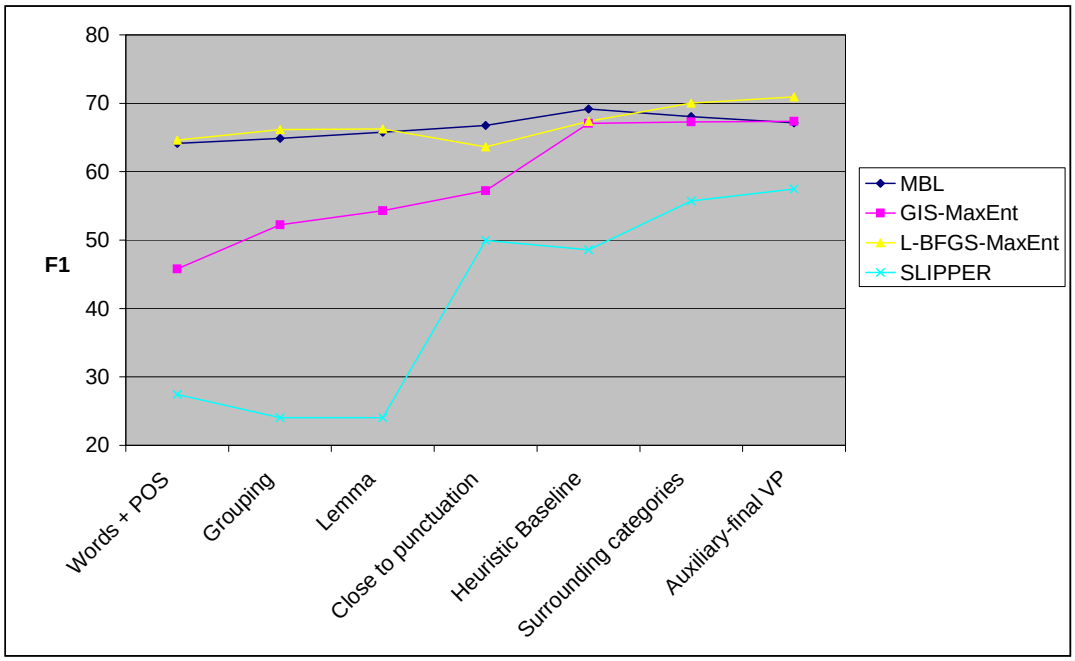

Figure D.16: F1 plot for algorithms on RASP parsed combined data versus features being added

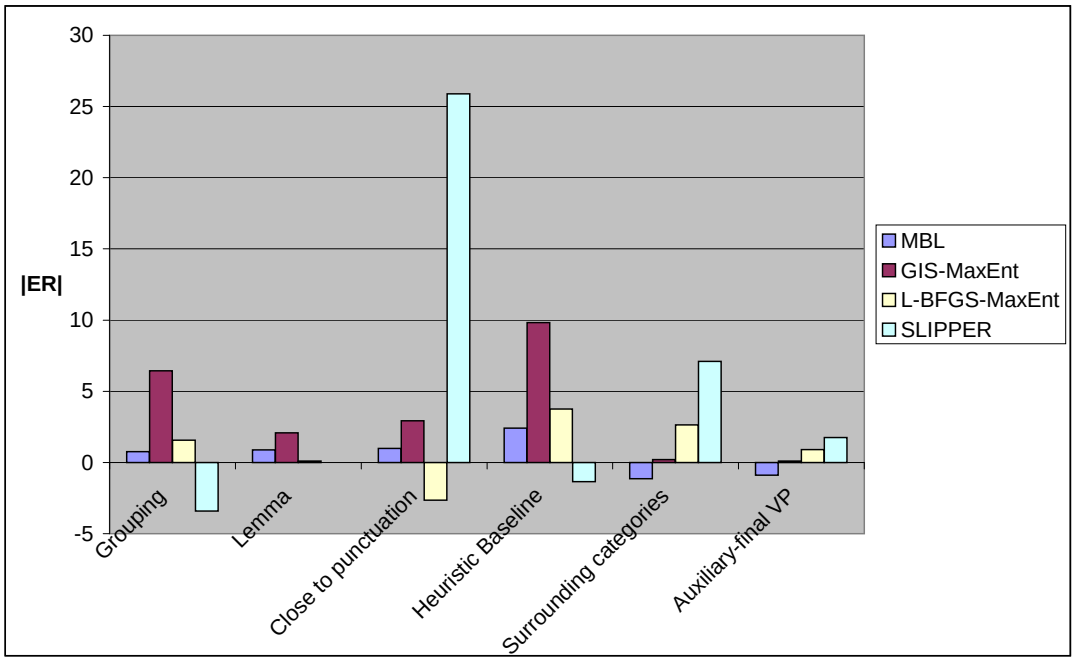

Figure D.17: Error Reduction effect of features on RASP parsed combined data

	Recall	Precision	F1	|ER|	%ER
Words + POS	20.56	41.24	27.44	-	-
+ group	21.13	27.88	24.04	-3.40	-4.69
+ lemma	21.13	27.88	24.04	0	0
+ close-to-punct	54.65	45.97	49.94	25.90	34.10
+ heuristic baseline	76.06	35.71	48.60	-1.34	-2.68
+ surrounding cat.	67.32	47.51	55.71	7.11	13.83
+ auxiliary-final VP	**72.68**	**47.51**	**57.46**	1.75	3.95

Table D.24: Results on combined data parsed with RASP - SLIPPER

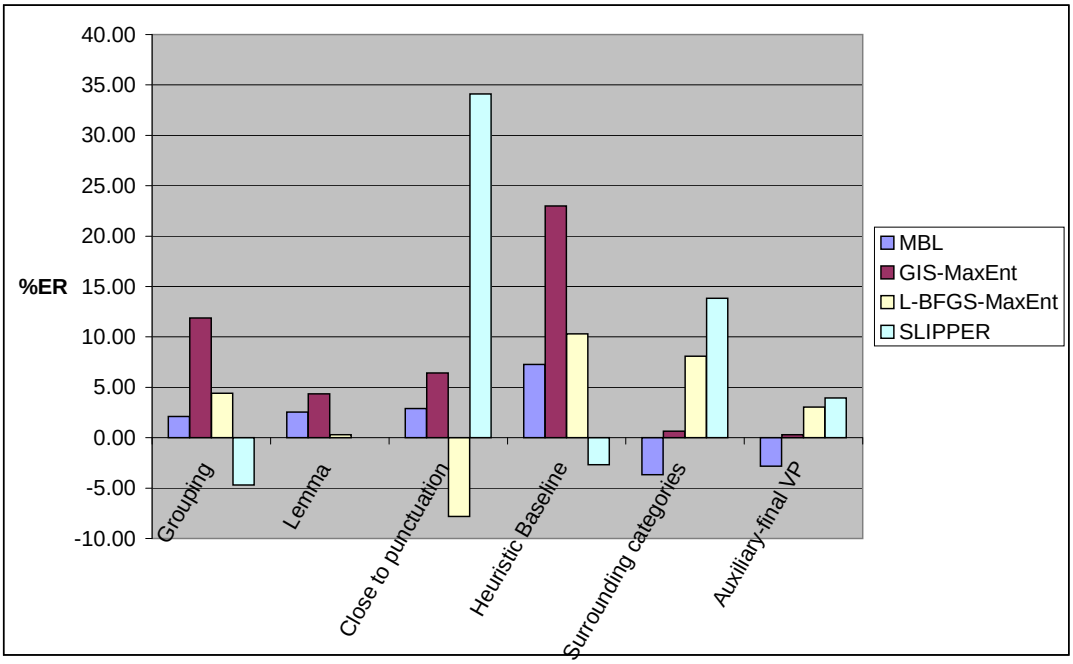

Figure D.18: Percentage Error Reduction effect of features on RASP parsed combined data

Appendix E

Detailed tables for antecedent location experiments

E.1 Information contributed by features

The statistics computed by TiMBL on the Treebank data is seen below.

Feats	Vals	InfoGain	GainRatio
1	(ignored)		
2	(ignored)		
3	20	0.21275143	0.049306709
4	259	0.25771564	0.034462515
5	17	0.14219737	0.041288852
6	2	0.041452831	0.29467811
7	2	0.0033015614	0.32274332
8	2	0.010258086	0.021843147
9	2	0.0033106271	0.0082957198
10	2	0.014235495	0.021253579
11	59	0.0093377921	0.0021421686
12	8	0.00092871998	0.00036414261
13	8	0.0060920949	0.0030189935
14	2	0.0012207778	0.0021960193
15	2	1.1462754e-05	1.1491989e-05

```
   16       5     0.0054978369      0.0035721699
   17       2     0.0015999122      0.14109133
   18     131     0.21984760        0.039990107

Feature Permutation based on GainRatio/Values :
< 7, 6, 17, 8, 10, 9, 3, 5, 14, 16, 13, 18, 4, 12, 11, 15, 1, 2 >
```

where the features are[1]

3	$Recency$
4	$Worddistance$
5	$Sententialdistance$
6	$SBARrelation$
7	$Comparativerelation$
8	$Auxiliarymatch$
9	$Be-domatch$
10	$In-quotes$
11	$Antecedentsize$
12	$VPEauxiliary$
13	$Antecedentauxiliary$
14	$As-appositive$
15	$Polarity$
16	$Adjuncts$
17	$Coordination$
18	$VPE-RESrank$

E.2 Rules learned by C4.5

```
Rule 181:
     Polarity = negative
     VPE-RES rank > 7
     VPE-RES rank <= 80
->  class FALSE  [99.9%]
```

[1]Features 1 and 2 are ignored, as they contain location information.

```
Rule 163:
     Sentential distance <= -1
->  class FALSE  [99.9%]

Rule 175:
     VPE auxiliary = be
     VPE-RES rank > 7
->  class FALSE  [99.9%]

Rule 177:
     Word distance <= -3
     VPE auxiliary = do
->  class FALSE  [99.8%]

Rule 182:
     Adjuncts = no_adjunct
     VPE-RES rank > 7
     VPE-RES rank <= 80
->  class FALSE  [99.8%]

Rule 170:
     VPE auxiliary = do
     Word distance > 6
     VPE-RES rank > 4
->  class FALSE  [99.7%]

Rule 66:
     As-appositive = as_appositive
     VPE-RES rank > 1
->  class FALSE  [99.7%]

Rule 43:
     VPE auxiliary = other
```

```
     Recency <= 1
     VPE-RES rank > 1
->  class FALSE  [99.6%]

Rule 171:
     VPE auxiliary = have
     VPE-RES rank > 4
->  class FALSE  [99.5%]

Rule 149:
     Sentential distance > 1
     VPE-RES rank > 2
->  class FALSE  [99.5%]

Rule 82:
     As-appositive = as_appositive
     Word distance > 10
->  class FALSE  [99.5%]

Rule 193:
     VPE auxiliary = can
     VPE-RES rank > 4
     VPE-RES rank <= 139
->  class FALSE  [99.5%]

Rule 99:
     Antecedent size <= 3
     Antecedent auxiliary = other
     Recency > 1
     Word distance > 10
     VPE-RES rank > 1
->  class FALSE  [99.3%]

Rule 32:
```

```
     Word distance <= 2
     Antecedent auxiliary = have
->  class FALSE  [99.2%]

Rule 90:
     Recency > 1
     Word distance > 15
->  class FALSE  [99.1%]

Rule 71:
     In-quotes = clashing
->  class FALSE  [98.9%]

Rule 98:
     Polarity = not_negative
     Recency > 1
     Word distance > 10
     VPE-RES rank > 1
->  class FALSE  [98.6%]

Rule 105:
     Antecedent auxiliary = would
     Word distance > 10
->  class FALSE  [98.4%]

Rule 5:
     VPE auxiliary = be
     Sentential distance > 0
     Antecedent auxiliary = other
     Polarity = negative
->  class FALSE  [98.4%]

Rule 106:
     Antecedent auxiliary = be
```

```
     Word distance > 10
     VPE-RES rank > 1
->  class FALSE  [98.2%]

Rule 63:
     VPE auxiliary = can
     Recency > 1
     Word distance > 6
     In-quotes = not_clashing
     VPE-RES rank > 1
->  class FALSE  [98.1%]

Rule 37:
     Antecedent auxiliary = to
     Recency <= 1
     In-quotes = not_clashing
     VPE-RES rank <= 2
->  class TRUE  [95.6%]

Rule 117:
     Sentential distance > 0
     In-quotes = not_clashing
     VPE auxiliary = do
     Word distance <= 6
->  class TRUE  [95.5%]

Rule 45:
     Recency > 1
     Word distance <= 4
     In-quotes = not_clashing
->  class TRUE  [95.0%]

Rule 52:
     In-quotes = not_clashing
```

```
     Antecedent size > 7
     Word distance <= 10
     VPE auxiliary = be
     VPE-RES rank <= 2
->  class TRUE  [92.2%]

Rule 21:
     In-quotes = not_clashing
     Word distance <= 10
     VPE-RES rank <= 1
->  class TRUE  [88.6%]

Rule 89:
     Word distance <= 15
     Antecedent size > 2
     VPE-RES rank <= 1
->  class TRUE  [87.8%]

Rule 64:
     SBAR relation = not_sbar_rel
     VPE auxiliary = should
     Word distance <= 10
     VPE-RES rank <= 2
->  class TRUE  [87.1%]

Rule 121:
     Recency > 1
     Word distance <= 6
     As-appositive = not_as_appositive
     VPE-RES rank <= 4
->  class TRUE  [86.9%]

Rule 77:
     SBAR relation = not_sbar_rel
```

```
     Antecedent auxiliary = be
     Recency <= 1
     VPE-RES rank <= 2
->  class TRUE  [85.7%]

Rule 79:
     Antecedent auxiliary = have
     As-appositive = not_as_appositive
     Recency <= 1
     VPE-RES rank <= 2
->  class TRUE  [83.8%]

Rule 183:
     VPE auxiliary = to
     Polarity = not_negative
     Adjuncts = ant_adjunct_only
     Word distance <= 19
     Sentential distance > -1
     VPE-RES rank > 7
->  class TRUE  [83.3%]

Rule 87:
     Adjuncts = no_adjunct
     VPE-RES rank <= 1
->  class TRUE  [82.2%]

Rule 131:
     In-quotes = not_clashing
     VPE auxiliary = would
     Word distance <= 11
     VPE-RES rank <= 4
->  class TRUE  [80.8%]

Rule 101:
```

```
     VPE auxiliary = do
     Word distance <= 15
     Sentential distance <= 1
     Antecedent size > 3
     As-appositive = not_as_appositive
     VPE-RES rank <= 2
->  class TRUE  [80.7%]

Rule 123:
     Polarity = negative
     Sentential distance <= 0
     As-appositive = not_as_appositive
     Word distance <= 11
     VPE-RES rank <= 4
->  class TRUE  [79.5%]

Rule 145:
     Word distance > 14
     In-quotes = not_clashing
     VPE auxiliary = would
     Word distance <= 15
->  class TRUE  [75.8%]

Rule 57:
     Word distance > 8
     SBAR relation = not_sbar_rel
     VPE auxiliary = do
     Antecedent auxiliary = other
     Recency > 1
     Word distance <= 10
     As-appositive = not_as_appositive
     VPE-RES rank <= 2
->  class TRUE  [70.7%]
```

```
Rule 132:
     In-quotes = not_clashing
     VPE auxiliary = to
     Word distance <= 11
     VPE-RES rank > 2
     VPE-RES rank <= 4
->  class TRUE  [70.0%]

Rule 107:
     Antecedent auxiliary = can
     As-appositive = not_as_appositive
     VPE-RES rank <= 2
->  class TRUE  [64.4%]

Rule 174:
     Adjuncts = vpe_adjunct_only
     Word distance <= 19
     VPE-RES rank > 4
     VPE-RES rank <= 7
->  class TRUE  [63.0%]

Rule 62:
     SBAR relation = not_sbar_rel
     VPE auxiliary = to
     Recency > 1
     Word distance <= 10
     In-quotes = not_clashing
     VPE-RES rank > 1
->  class TRUE  [61.1%]

Rule 154:
     In-quotes = clashing
     Antecedent size > 10
     VPE auxiliary = do
```

```
     Word distance > 15
     VPE-RES rank <= 4
->  class TRUE  [50.0%]

Rule 158:
     VPE auxiliary = other
     Antecedent auxiliary = be
     Adjuncts = ant_adjunct_only
     Word distance > 15
     As-appositive = not_as_appositive
     VPE-RES rank <= 4
->  class TRUE  [50.0%]

Rule 161:
     VPE auxiliary = other
     Adjuncts = vpe_adjunct_only
     VPE-RES rank > 2
     VPE-RES rank <= 3
->  class TRUE  [50.0%]

Rule 194:
     Sentential distance > -1
     VPE-RES rank > 139
->  class TRUE  [50.0%]

Default class: FALSE
```

E.3 Rules learned by SLIPPER

```
TRUE 0.028076 0 IF VPE-RES rank <= 7 Word distance <= 6 VPE-RES
rank >= 5 Recency <= 7 .
TRUE 0.0277479 0 IF VPE-RES rank >= 82 Sentential distance >= 0
VPE auxiliary = to .
```

```
TRUE 0.0214659 0 IF VPE-RES rank >= 60 Word distance >= -13 VPE-RES
rank <= 60 Antecedent size <= 4 .
TRUE 0.0157358 0 IF Antecedent auxiliary = have Word distance <= 18
Word distance >= -3 VPE-RES rank >= 2 VPE-RES rank <= 36 VPE-RES
rank >= 6 Word distance <= 9 .
TRUE 0.0305465 6.50546e-05 IF Antecedent size >= 28
Sentential distance >= 0 Word distance <= 6 VPE-RES rank <= 41 .
TRUE 0.0117885 3.68939e-05 IF Auxiliary match = aux_match
Antecedent size <= 4 Antecedent size >= 2
SBAR relation = not_sbar_rel VPE-RES rank <= 3 Antecedent size <= 3
Recency >= 5 Sentential distance <= 2 .
TRUE 0.00640428 0 IF VPE-RES rank >= 140 Sentential distance >= 0 .
TRUE 0.0334545 0.000428514 IF VPE auxiliary = to Word distance <= 7
Polarity = not_negative Recency >= -7 Adjuncts = ant_adjunct_only
VPE-RES rank <= 46 VPE-RES rank >= 36 In-quotes = not_clashing .
TRUE 0.296091 0.00495414 IF VPE-RES rank <= 1 .
TRUE 0.00417111 2.68476e-05 IF Antecedent auxiliary = should
Recency <= 2 VPE-RES rank <= 3 .
TRUE 0.00276764 0 IF Comparative relation = comp_rel .
TRUE 0.061269 0.00208891 IF VPE auxiliary = to
Adjuncts = ant_adjunct_only Sentential distance >= 0 Recency <= 5
VPE-RES rank >= 4 In-quotes = not_clashing .
TRUE 0.0319429 0.00115595 IF Antecedent auxiliary = can
Antecedent size <= 3 Sentential distance >= 0 Antecedent size >= 1
VPE-RES rank >= 2 Word distance <= 58 As-appositive = not_as_appositive .
TRUE 0.137985 0.00677343 IF VPE-RES rank <= 3 Word distance <= 10
VPE-RES rank >= 2 In-quotes = not_clashing
As-appositive = not_as_appositive .
TRUE 0.0594982 0.00307488 IF VPE auxiliary = other Word distance >= 13
Word distance <= 36 VPE-RES rank >= 2 As-appositive = not_as_appositive
In-quotes = not_clashing .
TRUE 0.00705622 0.000542679 IF Coordination = coordinated .
TRUE 0.0559312 0.0059628 IF VPE auxiliary = would VPE-RES rank <= 6
VPE-RES rank >= 2 Sentential distance <= 1 .
```

```
TRUE 0.0554276 0.00600499 IF VPE auxiliary = to VPE-RES rank >= 38 .
TRUE 0.082696 0.0100996 IF VPE-RES rank <= 6 Antecedent size >= 9
Polarity = negative VPE-RES rank >= 4 .
TRUE 0.067907 0.00997801 IF Antecedent auxiliary = to Word distance <= 25
Word distance >= 3 Auxiliary match = not_aux_match Antecedent size <= 11 .
TRUE 0.0215372 0.00345079 IF Antecedent auxiliary = have VPE-RES rank <= 3
Antecedent size >= 6 .
TRUE 0.0461894 0.0079856 IF VPE-RES rank <= 6 Word distance <= 19
VPE-RES rank >= 5 .
TRUE 0.0944434 0.0174581 IF Auxiliary match = aux_match
Sentential distance >= 0 Recency <= 5 In-quotes = not_clashing
Antecedent size <= 10 Word distance <= 39 Word distance >= -13
Sentential distance <= 4 As-appositive = not_as_appositive
SBAR relation = not_sbar_rel .
TRUE 0.344267 0.0637989 IF VPE-RES rank <= 4 Word distance <= 15
Word distance >= 2 .
TRUE 0.0259948 0.00616658 IF Antecedent auxiliary = do
Antecedent size >= 5 .
TRUE 0.15818 0.038227 IF VPE-RES rank <= 6 Word distance <= 6 .
TRUE 0.98704 0.314036 IF VPE-RES rank <= 4 .
TRUE 0.236384 0.0765126 IF VPE-RES rank <= 5 Word distance <= 26
Antecedent size >= 3 As-appositive = not_as_appositive .
TRUE 0.217715 0.115488 IF VPE auxiliary = do Sentential distance >= 0 .
TRUE 0.390204 0.219782 IF VPE-RES rank <= 7 Word distance <= 26 .
TRUE 0.0434998 0.0254557 IF Adjuncts = vpe_adjunct_only .
TRUE 0.313157 0.198627 IF VPE-RES rank <= 5 Word distance <= 13 .
TRUE 0.493557 0.367963 IF Sentential distance >= 0 .
TRUE 0.500059 0.499959 IF .
FALSE 0.000145057 0.924811 IF .
```

Appendix F

Append or replace

While simple appending is taken as the default, there's more to be said about whether the VPE auxiliary is kept or replaced by the antecedent. For the most part, this is determined by grammatical rules and parallelism :

(79) a. I looked just as ridiculous as you did.

b. ? I looked just as ridiculous as you *did look*.

c. I looked just as ridiculous as you *looked*.

(80) a. The room seemed darker now, and smaller than it had.

b. * The room seemed darker now, and smaller than it *seemed*.

c. The room seemed darker now, and smaller than it *had seemed*.

(81) a. However, this seems to be a red herring, since the standard Buck topology uses a pair of inductors (though not coupled) and a single transistor switch just as the Cuk topology does, but need no other energy coupling device.

b. ? However, this seems to be a red herring, since the standard Buck topology uses a pair of inductors (though not coupled) and a single transistor switch just as the Cuk topology does *use a pair of inductors*

(though not coupled) and a single transistor switch, but need no other energy coupling device.

c. However, this seems to be a red herring, since the standard Buck topology uses a pair of inductors (though not coupled) and a single transistor switch just as the Cuk topology *uses a pair of inductors (though not coupled) and a single transistor switch*, but need no other energy coupling device.

It should be noted that while the resolved versions of the sentences may not sound natural, this is to be expected, given that ellipsis was employed in the first place to avoid repetition.

There are cases where both appending to the auxiliary and replacing it are grammatical and sensible :

(82) a. I had read the story many times without asking myself why it affected me or caring why it did.

b. I had read the story many times without asking myself why it affected me or caring why it *affected me*.

c. I had read the story many times without asking myself why it affected me or caring why it did *affect me*.

(83) a. I didn't look.

b. GUIL : Yes you did.

c. GUIL : Yes you *looked*.

d. GUIL : Yes you did *look*.

In (82), the emphasis on the words could suggest a preferred resolution; a stress on *why* would suggest (82b) more, while a stress on *did* would suggest (82c). In (83), it is difficult to choose one resolution over the other.

In other instances, contrast can suggest a preferred reading :

(84) a. There was no question about it - people knew who I was and if they didn't they asked and I told them.

b. GUIL : You did, the trouble is, each of them is plausible, without being instinctive.

c. GUIL : ? You *told them*, the trouble is, each of them is plausible, without being instinctive.

d. GUIL : You did *tell them*, the trouble is, each of them is plausible, without being instinctive.

(85) a. "Ah," Mr Starke said, sitting forward in his seat and looking more interested, "you resigned".

b. "I certainly did."

c. ? "I certainly *resigned*."

d. "I certainly did *resign*."

In most cases grammatical rules determine whether to append to the VPE or replace it, while in other cases, the choice depends on discourse related information, such as contrast, emphasis etc. To be able to choose between the two reliably, a dialogue tracking system would be necessary.