Corpus Annotation

Corpus Annotation

Linguistic Information from Computer Text Corpora

Roger Garside Geoffrey Leech
Tony McEnery

Editors

Routledge
Taylor & Francis Group
LONDON AND NEW YORK

First published 1997 by Addison Wesley Longman Limited

Published 2013 by Routledge
2 Park Square, Milton Park, Abingdon, Oxon OX14 4RN
711 Third Avenue, New York, NY 10017, USA

Routledge is an imprint of the Taylor & Francis Group, an informa business

Notices
Knowledge and best practice in this field are constantly changing. As new research and experience broaden our understanding, changes in research methods, professional practices, or medical treatment may become necessary.

Practitioners and researchers must always rely on their own experience and knowledge in evaluating and using any information, methods, compounds, or experiments described herein. In using such information or methods they should be mindful of their own safety and the safety of others, including parties for whom they have a professional responsibility.

ISBN 13: 978-0-582-29837-8 (pbk)

British Library Cataloguing-in-Publication Data

A catalogue record for this book is
available from the British Library

Library of Congress Cataloging-in-Publication Data

Corpus annotation: linguistic information from computer text corpora
/ Roger Garside, Geoffrey Leech, Tony McEnery, editors
p. cm.
Includes bibliographical references and index.
ISBN 0–582–29837–7 (paper : alk. paper)
1. Computational linguistics. 2. Linguistics—Notation. I. Garside, Roger.
II. Leech, Geoffrey N. III. McEnery, Tony, 1964–
P98.C6376 1997
410'.285—dc21 97–15523 CIP

Contents

Contributors

JOHN PAUL BAKER
Lancaster University

RUTHANNA BARNETT
Lancaster University

JEREMY BATEMAN
Lancaster University

SIMON BOTLEY
Lancaster University

ELIZABETH EYES
Lancaster University

STEVE FLIGELSTONE
Lancaster University

JEAN FORREST
Lancaster University

ROGER GARSIDE
Lancaster University [Editor]

JOHN HUTCHINSON
Lancaster University

PETER KAHREL
Lancaster University

JEAN-MARC LANGÉ
IBM Speech Business Unit, Paris

GEOFFREY LEECH
Lancaster University [Editor]

TONY MCENERY
Lancaster University [Editor]

AMALIO F. NIETO SERRANO
Ciudad Universitaria s/n

MICHAEL OAKES
Lancaster University

MIKE PACEY
Liverpool University

PAUL RAYSON
Lancaster University

FERNANDO SÁNCHEZ LEÓN
Universidad Autónoma de Madrid

NICHOLAS SMITH
Lancaster University

JENNY THOMAS
University College of North Wales, Bangor

JEAN VÉRONIS
Université de Provence & CNRS

TIM WILLIS
Lancaster University

ANDREW WILSON
Lancaster University

MARTIN WYNNE
Lancaster University

Preface

It is time to take stock of developments in a relatively new area of research within the field of computational linguistics. The terms 'corpus enrichment' and 'corpus analysis' have been used for this area, but the term **corpus annotation** appears to be the current favourite. To understand what this means, the newcomer to this area may find it helpful to think in terms of stages of development:

Step 1 A corpus (or body) of texts in some language is collected and stored electronically, on computer. The selection of texts (whether written or spoken) could be of a particular variety or register of language, or the goal could be a comprehensive sample of as wide a range of language varieties as possible.

Step 2 Someone wishes to extract information from this corpus: say, to create better dictionaries or grammars of the language from 'real language data', so that these can be used for understanding and manipulating the language more successfully. The ultimate objectives of this stage of operation may vary – but today, the development of better communication between humans and machines must be among the most important.

Step 3 In order to extract information from corpuses (or, to use the more learned Latin plural, *corpora*) it is found necessary first to add more explicit linguistic information to the texts. We have to analyse the corpus linguistically at one or more levels, and annotate or 'label' the corpus with the information thus obtained.

The present book focuses on the details of this step. Since the first annotated corpus was produced in 1978, interest and activity in corpus annotation has grown stage by stage, so that now work is progressing and diversifying at a considerable speed. This is the first book to be published on the topic in general. All that can be attempted here is a still photograph, so to

speak, of where we had reached when this book was being assembled, at the beginning of 1997.

In design, this book divides into three parts. **The first part** (Chapters 1–6) covers the nature of the linguistic annotation of corpora, leading from an introduction (Chapter 1) to the various levels of annotation that may be applied to a corpus. Of these, the most common and successful level is grammatical word-class tagging (Chapter 2), followed by syntactic annotation (Chapter 3). In the subsequent three chapters we turn to levels of linguistic annotation where progress has been somewhat less advanced, largely because of the more abstract and/or difficult nature of the phenomena to be analysed: semantic annotation (Chapter 4), discoursal annotation (Chapter 5), and prosodic, pragmatic and stylistic annotation (Chapter 6).

In **the second part** (Chapters 7–12), the focus moves from annotation *per se* to the process of annotating text corpora: the kinds of software in use or under development, and the relation between machine processing and human processing as contributors to this overall task. Again, different chapters are devoted to different levels of annotation: Chapters 7–10 deal mainly with grammatical word-class tagging, Chapter 11 with syntactic annotation, and Chapter 12 with other levels.

The third part of the book (Chapters 13–17), building on ground laid in the previous parts, takes in a range of wider perspectives. Chapter 13 ('A corpus/annotation toolbox') looks at the overall software environment of corpus annotation and annotated corpora. Chapters 14 and 15 examine two examples of applications (educational and multilingual) of annotated corpora. Finally Chapters 16 and 17 survey issues of annotation standards and the evaluation of annotative practices.

To conclude, there are three appendices. In Appendix I will be found sources of further information, such as email addresses and World-Wide Web sites; Appendix II provides a list of software names, acronyms and other abbreviations used in the book; and Appendix III supplies selected lists of grammatical tags.

With some notable exceptions, the contributors to this volume are members of the UCREL research team at Lancaster University. UCREL (University Centre for Computer Corpus Research on Language) has been engaged in corpus annotation over a period of nearly two decades, and has

extended its work in this field to cover many different kinds of annotation practices. (*Any reader interested in acquiring copies of software produced by* UCREL, *generally only for internal academic, non-commercial use, should contact* UCREL *for up-to-date information, by email at ucrel@lancaster.ac.uk.*) However, the book is not merely a record of one team's efforts: it also surveys international developments, and pays attention to other centres, such as those at the University of Pennsylvania and at the University of Helsinki, which have made leading contributions to advances in corpus annotation.

In one respect, this book must be acknowledged as having a limited perspective. The language on which most work in corpus annotation has so far been carried out is English: for this reason, and for reasons of general intelligibility, English is also the language on which this book concentrates. Other languages are represented only in Chapter 10 (which deals with Spanish) and, to some extent, in the multilingual perspectives of Chapters 15 and 17. While much work on corpus annotation is now being undertaken for the first time in a wide range of languages, it would perhaps have been premature to give full attention to the multilingual dimension of corpus annotation at the present time. In three to five years from now, there will no longer be any excuse for allowing English the dominant place it holds in this book. But, for those who have primary interests in other languages, we hope that the focus in some depth on one language as a 'case study' will have some transferable benefit.

In preparing this book for publication, we owe a debt to the following, whose help has been much appreciated. Peter Kahrel and Ruthanna Barnett have given us expert and meticulous support as technical editors. Ezra Black of ATR (Kyoto) has given careful attention to Chapter 11, vetting the content and also facilitating permissions. As a final check, Andrew Wilson has read the whole book in proof. To all these, as well as to the other contributors and to the editorial staff of Longman, we give our sincere thanks.

Lancaster, July 1997

Roger Garside
Geoffrey Leech
Tony McEnery

1

Introducing Corpus Annotation

GEOFFREY LEECH

1.1 What is a Corpus and What is Corpus Annotation?

Traditionally, linguists have used the term **corpus** to designate a body of naturally-occurring (**authentic**) language data which can be used as a basis for linguistic research. This body of data may consist of written texts, spoken discourses, or samples of spoken and/or written language. Often it is designed to represent a particular language or language variety. In the past thirty-five years, the term **corpus** has been increasingly applied to a body of language material which exists in electronic form, and which may be processed by computer for various purposes such as linguistic research and language engineering (see Leech 1991, Leech and Fligelstone 1992, Church and Mercer 1993, McEnery and Wilson 1996). As the power and capacity of computers have increased, corpora have increased dramatically in size, variety and ease of access. At the same time, an increasing range of software has been developed to process corpora and access the information they contain. A computer corpus is fast becoming a universal resource for language research on a scale unimaginable thirty-six years ago.

The mention of a period of thirty-six years is not fortuitous. The year 1961, which more famously saw the first manned space flight, is the date to which corpus linguists can look back as the date when the enterprise now known as **corpus linguistics** (or more precisely **computer corpus linguistics**) came into being. This was the date when work began on the first electronic corpus, later to be known as the Brown Corpus[1] (after Brown University, Providence, RI, where the corpus was compiled). The corpus consisted of just over one million words, comprising 500 text samples of about 2,000 words each. The samples were all taken from publications in the year 1961, and the corpus was complete and ready for distribution on magnetic tape in 1964. As an indication of how the size of corpora has increased since 1964, the one-million-word Brown Corpus

seems small today beside the corpus products of the 1990s, including the 100-million-word British National Corpus (BNC),[2] completed in 1994 and containing 10 million words of transcribed speech, and the even larger Bank of English, which runs to more than 300 million words.[3]

However, the value of a corpus as a research tool cannot be measured in terms of brute size. The **diversity** of the corpus, in terms of the variety of registers or text types it represents, can be an equally important (or even more important) criterion. So, too, can the care with which it has been compiled, for example, with respect to the faithful encoding of orthographic features of the text. A fourth factor, the degree to which 'added value' is brought to a corpus by **annotation**, is the subject of this book. Corpus annotation is widely accepted as a crucial contribution to the benefit a corpus brings, since it enriches the corpus as a source of linguistic information for future research and development. Further, as this book will aim to demonstrate fully, corpus annotation has become an important and fascinating area of research in its own right.

But what *is* corpus annotation? It can be defined as the practice of adding **interpretative**, **linguistic** information to an electronic corpus of spoken and/or written language data. 'Annotation' can also refer to the end-product of this process: the linguistic symbols which are attached to, linked with, or interspersed with the electronic **representation** of the language material itself. A typical and familiar case of corpus annotation is **grammatical tagging** (also called word-class tagging, part-of-speech tagging or POS tagging). In this case, a label or **tag** is associated with a word (e.g. by some kind of attachment symbol such as the underline character or the slash character), to indicate its grammatical class: for example, in *taken*_VVN, the grammatical tag VVN shows that *taken* is a past participle. The definition of annotation above, and in particular the use there of the terms 'interpretative' and 'linguistic', requires some further discussion.

First, by calling annotation 'interpretative', we signal that annotation is, at least in some degree, the product of the human mind's understanding of the text. There is no purely objective, mechanistic way of deciding what label or labels should be applied to a given linguistic phenomenon.[4] Disagreement is unlikely to occur if we label *taken* as a past participle – which is conventionally the grammatical class it belongs to in English. But

Box 1.1 Example of grammatical tagging, using the C5 tagset of the BNC

```
High_AJ0 winds_NN2 and_CJC heavy_AJ0 seas_NN2 have_VHB been_VBN
causing_VVG further_AJ0 problems_NN2 in_PRP the_AT0 southern_AJ0
part_NN1 of_PRF Britain_NP0 ,_PUN leaving_VVG homes_NN2
flooded_VVN ,_PUN and_CJC roads_NN2 blocked_VVN ._PUN
```

there are many other words which would be more contentious: for example: *future* in *his future bride.* Is it a noun or an adjective? Or, to take up a question of how much detail (**delicacy** or **granularity** are the terms often used for 'detail') should be encoded through annotation, if *future* is an adjective, should it be labelled as an adjective of a particular subclass – say, the class of adjectives which must occur in a pre-nominal position? (We can say *his future bride*, but not **His bride will be future.*) Decisions about these and many other matters have to be taken when we set out to annotate a corpus (see below).

Second, we assume a distinction between the 'annotation' and 'representation' of a text – a distinction which may be easy or less easy to apply. For a written text, generally these two kinds of information are relatively easy to separate. The purely orthographic record of a text is a sequence of written characters from (say) the Roman alphabet, interspersed with spaces and punctuation marks (with occasional use of visual material, numerals and 'non-standard' characters such as mathematical symbols). This record can be represented electronically in the computer by special codes and mark-up,[5] and a one-to-one mapping between these and visual symbols can be maintained: the original orthographic document is simply replaced by an unambiguous representation in the form of an electronic document. It is true that some more or less detailed information may be lost in this process – e.g. font and type-size may no longer be retrievable – but this is felt to be allowable if such information is not judged to be essential to the representation of the text as a linguistic phenomenon. In contrast to this, the *annotation* of a text is *meta*linguistic: instead of telling us what the text itself comprises,[6] it gives information *about* the language of the text.

For a spoken discourse, however, it is not easy to distinguish between representational and interpretative information. In rendering speech in written or electronic form (except where the representation is purely instrumental, as in the case of acoustic wave forms), a transcriber must necessarily *interpret* the discourse in the course of representing it. Most transcriptions, as a matter of convenience, incorporate conventionally-spelt words, using phonetic transcription, if at all, only for exceptional pronunciation. But this merely gives superficial readability to speech events whose real nature – physical, linguistic, or social – may be vastly more complex and elusive. Prosodic labelling of stress and intonation, for example, is to some extent dependent on the judgement and expertise of the transcriber (Knowles 1991), as well as on the system of analysis adopted. There is no doubt that prosodic labelling at one level is a **representation** of part of the data of the speech event being transcribed. However, there is equally no doubt that prosodic labelling is in part an **interpretation** of the event

through the auditory perception of this or that listener, even where the perceiver is a highly trained phonetician (Pickering *et al.* 1996).[7] For the purposes of this book, we have decided to give some attention to phonetic and prosodic annotation as types of annotation, while acknowledging their in-between status.[8]

Although the distinction between the **raw corpus** (some prefer 'pure corpus') and the interpretative annotations can be somewhat artificial, it is nevertheless a useful distinction, since we should not see annotations as having the claim to reality and authenticity which belongs to the corpus itself. For a written corpus, the text itself is the data (in the etymological sense **data** are 'givens'), and the annotations are superimposed on it. For a spoken corpus, the recording is what is 'given', and it can also be maintained that a bare verbatim transcript of 'what was said' is itself a kind of 'secondary given', that is, a written record without any addition of less reliable, less clearly-definable, information.[9] Beyond these 'givens' it is difficult to go without implicitly taking up some descriptive or interpretative stance towards the data.

1.2 Why Annotate a Corpus?

Why is it important to be able to annotate a corpus?

1.2.1 Extracting information

Corpora are useful only if we can extract knowledge or information from them. The fact is that to extract information from a corpus, we often have to begin by building information in – that is, by adding annotations. The 'raw corpus' in its orthographic form contains no direct information, for example, about grammar – and this can hinder many of the applications to which a corpus can be put. Consider the word spelt *left*. As a word meaning the opposite of *right*, it can be an adjective ('my *left* hand'), an adverb ('turn *left*') or a noun ('on your *left*'). As the past tense or past participle of *leave*, it is a verb ('I *left* early'). *Left* is therefore a very versatile piece of language – but its various meanings and uses cannot be detected from its orthographic form. This is a disadvantage for one of the most salient uses of a corpus in recent years – its use as a resource for lexicography. But if a corpus is successfully grammatically tagged, each occurrence of *left* will be accompanied by a label indicating its word-class. This is a pre-requisite for anyone using a corpus for making or improving dictionaries. To take another example: the word spelt *lead* in English can be either a noun, pronounced /led/, or a verb, pronounced /li:d/. If we

want to create a machine for converting written language into auditory 'spoken' output – a **speech synthesizer** – it is necessary for the synthesizer to distinguish the noun from the verb, if it is to produce a correct pronunciation. Once again, a grammatically tagged corpus would provide the synthesizer with the information it needs.

1.2.2 Re-usability

It might be argued that to extract information of the types mentioned above, there is no need for an exhaustive annotation of a corpus. It might be sufficient to run a clever little program to recognize that, for example, *left* preceding a noun is an adjective, or that *left* following a verb is an adverb. Such little programs could run 'on the fly' extracting instances of the target word without undertaking any annotation. However, such an argument has two weaknesses. First, it is evident from the example of *left* that, in order to identify the word-class of the target word, we would also have to presuppose knowledge of the word-class of neighbouring words. In other words, the identification of word-classes (or any other linguistic phenomena) cannot be treated as an isolated problem. Second, the point about grammatical tagging and other levels of annotation is: once the annotation has been added to the corpus, the resulting annotated corpus is a more valuable resource than the original corpus, and can now be handed on to other users. This argument of 're-usability' is a powerful one, since corpus annotation tends to be an expensive and time consuming business. We do not want to waste resources by 're-inventing the wheel' time and time again – i.e. by re-analysing or re-annotating the same corpus material. An annotated corpus, like any corpus, is valuable because it is a re-usable resource.

1.2.3 Multi-functionality

Taking the point about re-usability one step further, we may note that annotation often has many different purposes or applications: it is multifunctional. We have already noted the application of grammatical tagging to the two different applications of lexicography and speech synthesis. Other language engineering applications – such as machine-aided translation or information retrieval – could also be mentioned. But the general point to make is that annotation gives 'added value' to a corpus in the general sense: it adds overt linguistic information, which can then be used for a multitude of purposes. Thus grammatical tagging is often considered a kind of 'base camp' annotation which can be the first step towards more difficult levels of annotation such as those of syntax and semantics. The

reusability of annotated corpora is enhanced by the fact that there are many different purposes for which others may wish to make use of the annotations: purposes which the original annotators of the corpus may not even have thought of.

1.3 Some Standards for Corpus Annotation

Our acceptance of annotations as useful and informative must depend to a considerable extent on our evaluation of the 'experts' who added them to the corpus, and of the usefulness of the annotative scheme they have adopted. In the short history of corpus annotation, it has been by no means unusual for the builders of a corpus to add to it annotations which others have found difficult or impossible to use. To avoid this situation, we suggest that a number of practical guidelines, or standards of good practice, should apply to any project for annotating corpus texts:

1. It should always be possible, and easy, to dispense with the annotations, and to revert to the raw corpus. The raw corpus should be **recoverable**.
2. The annotations should, correspondingly, be **extricable** from the corpus, to be stored independently if there is a need.
3. The user of the corpus[10] should have (easy) access to **documentation**, which will include information on
   (a) The **annotation scheme** – that is, a document describing and explaining the scheme of analysis employed for the annotations.[11]
   (b) **How**, **where** and **by whom**, the annotations were applied.
   (c) Further, since annotations (given the typical size of annotated corpora) quite often contain erroneous, inconsistent or ambiguous elements, there should be some account of the **quality** of annotation: e.g. to what extent has the corpus been checked, what is its accuracy rate (e.g. the percentage of annotations which are judged correct), and to what extent is the application of annotations consistent (see Chapter 17).

On a more philosophical level, the following additional maxims apply generally both to the compilers and users of annotated corpora:

4. For reasons already given, there can be no claim that the annotation scheme represents 'God's truth'. Rather, the annotation scheme is made available to a research community on a *caveat emptor* principle. It does not come with any 'gold standard' guarantee, but is offered as a matter of practical usefulness only, on the assumption that many

users will find it valuable to use a corpus with annotations already built in, rather than to have to devise and apply their own annotations and annotation schemes from scratch (a task which could take years to accomplish).

5. Therefore, to avoid misunderstandings and misapplications, it is good idea for annotation schemes to be based as far as possible on **consensual** or theory-neutral analyses of the data. Perhaps the best analogy here is to the kind of structural or classificatory information given in printed dictionaries. A dictionary gives information about the grammatical classification of words, for example, but tends to take these as given by general descriptive traditions, rather than as coming from some theoretical model that has to be justified. While annotators are bound to face some theoretically sensitive decisions, their goal[12] should be to adopt annotations which are as widely accepted and understood as can be managed. (Perhaps it should be added that the existence and content of 'consensual categories' is not itself a matter on which it is easy to gain a consensus!)
6. No one annotation scheme should claim authority as an absolute **standard**. Annotation schemes tend to vary for good practical reasons. For example, the size of the corpus to be annotated may militate against too much detail. The purpose for which the annotations are primarily intended may give priority to certain kinds of information (e.g. a corpus which has been grammatically tagged mainly as a preliminary to parsing may need careful discriminations to be made between different kinds of subordinating or coordinating conjunction). The kind of corpus data (e.g. spoken vs. written) or the identity of the language (e.g. Chinese vs. Greek) may also encourage differences in the annotations to be applied.

Yet, in spite of (6) above, there is much to said in favour of some kind of standardization of corpus annotation practices, and it is likely that convergence towards some degree of uniformity of practice will take place in the next few years – indeed this convergence has already begun. One reason for standardization is inertia: if you are familiar with some annotation scheme that you have found useful (say, the Penn tagset for grammatical tagging, developed at the University of Pennsylvania – Santorini 1990), it makes sense to stick to that one in developing your own annotated corpus. Another reason is the already-emphasized principle of re-usability. If different researchers need to interchange data and resources (such as annotated corpora), this is more easily achieved if the same standards or guidelines have been applied in different centres. The need for some kind of standardization of annotation practices is particularly evident when we

come to the mutual exchange of corpus software utilities (see Chapter 13). Authorities who fund research may also find it desirable to exert influence in the direction of standardization: this has been recently noticed in the policy of the European Union in setting up the EAGLES initiative (see Chapter 16). But the need is to encourage convergent practice without imposing a straitjacket of uniformity which would inhibit flexibility and productive innovation.

1.4 A Glance at the History of Corpus Annotation

1.4.1 Beginnings of grammatical word tagging

To our knowledge, the first computer corpus annotation project to be undertaken was the word-class tagging of the Brown Corpus. Under the supervision of the founders of computer corpus linguistics, Francis and Kučera, this was undertaken by two M.A. students at Brown University, Greene and Rubin (1971), using a tagset of 77 different word-class labels. This was soon after the completion of the Brown Corpus itself. As may be supposed, such a large list of word-class tags would identify not only major parts of speech (noun, verb, preposition, etc.) but also values defining subclasses, such as singular and plural nouns, positive, comparative and superlative adjectives, and so on.

The outcome of this pioneering experiment was that 77 per cent of the words were successfully tagged and disambiguated. (For further discussion see Section 7.1) There still remained the considerable task, undertaken at Brown in the following years, of eliminating all 230,000 of the remaining ambiguities by manual editing of the corpus (see Francis 1980).

The experiment of Greene and Rubin eventually led to an extremely useful product: the word-class tagged Brown Corpus, which has since been used by many thousands of researchers all over the world. But the interest of the TAGGIT method of tagging is that it helps to identify, even at this pioneering stage, a number of general characteristics of corpus annotation. One distinction often made is between automatic and manual annotation of a corpus. Greene and Rubin found it necessary to adopt an automatic tagging method, but the completion of their task was a tedious and time-consuming manual editing of the whole corpus. This division of labour between automatic and manual methods is a recurring theme of corpus annotation, with a number of variations. Beyond a given corpus size (depending on the speed and complexity of annotation), purely manual methods are impracticable. At the other end of the scale, purely automatic annotation can only be tolerated if the result of the annotation is good

enough to use as it is: i.e. the error rate or ambiguity rate should be sufficiently low – no more than *n* per cent, where *n* is a small number, dependent on the application – to make the annotated corpus practically useful.

A second major tagging project, in 1979–82, was the tagging of the British counterpart of the Brown Corpus – the LOB Corpus[13] (Marshall 1983; Garside *et al.* 1987: Chapters 3–5). This time the tagging software employed probabilistic methods. Those tagging the LOB Corpus were fortunate enough to be able to use the tagged Brown Corpus as input, especially as a source of tag transitional frequency data. The success rate of automatic tagging leaped from 77 per cent to 96.7 per cent. However, a consequence of the probabilistic method was that the tagger (CLAWS1) inserted the most likely tag in every position, so that wherever it failed, it made errors. That is, 3.3 per cent of the tags were erroneous, and had to be corrected (not merely disambiguated) by hand.

After CLAWS1, a number of word-class taggers were devised, many of them using probabilistic methods (e.g. the taggers of Church (1988) and DeRose (1991)). A number of themes which recur in corpus annotation made their appearance in the decade following the LOB tagging project: the choice between probabilistic and non-probabilistic methods is still a bone of contention. Also, as the LOB tagging project shows, a probabilistic model requires a **training corpus**, a corpus preferably already annotated which supplies initial estimates on the basis of which the probabilistic annotation software is trained. In the case of CLAWS1, this was generously supplied by Kučera and Francis, the authors of the previously tagged Brown Corpus. Another interesting observation is that both TAGGIT and CLAWS, in spite of their different methods of tagging, used a very limited context (one or two words to the left or to the right) to determine the correct tag for a word. This, again, is a recurring issue of corpus annotation software: how far can we get by using extremely local information as a basis for automatic annotation? A final thing to note is that both TAGGIT and its successor CLAWS1 operated on the English language only. For a long time, and indeed up to about 1988, very little annotation of corpora for other languages took place, largely, no doubt, because such corpora did not exist.[14] Since 1990, however, the annotation of corpora has extended to many other languages (e.g. Chinese, Japanese, French, German, Polish, Spanish), and there has even been a move toward the development of language-independent corpus annotation software (especially Cutting *et al.*'s 1992 Xerox Parc tagger – see Chapter 10, especially Section 10.2).

A boom in grammatical tagging began in about 1987, and since that time many taggers have been developed for different languages. Now, however, it is time to backtrack to the mid-seventies to trace the development of other levels of annotation.

1.4.2 Beginnings of prosodic annotation

Since written corpora are easier to collect and compile than corpora of spoken discourse, it was not until the mid-1970s that a first major attempt was made to establish a computer corpus of spoken language. This was the London-Lund Corpus (LLC), which was in fact a computer-readable version of spoken materials from the Survey of English Usage corpus (eventually 500,000 words), which had been compiled in paper form at University College London from 1960.[15] The name 'London-Lund' derives from the fact that the computerization was undertaken at Lund, in Sweden (see Svartvik 1990). The LLC was also the first electronic corpus to have prosodic annotation/transcription built into it. The stress, intonation, pauses and other prosodic features had been transcribed in great detail over the preceding 15 years or so in London (see Peppé 1995). Another landmark worth mentioning was the completion in 1986 of the Lancaster/IBM Spoken English Corpus (SEC), which, although much smaller than the LLC, combined different levels of annotation within the same corpus: the same spoken texts were provided with grammatical tagging, syntactic annotation and prosodic annotation, as well as with co-existing orthographic and digitally-recorded versions.[16]

1.4.3 Beginnings of syntactic annotation

The mention of the syntactically annotated version of the SEC brings us to another part of the annotation story: the development of corpora with syntactic annotation. In the early days of electronic corpora, a pioneering effort by Ellegård (1978) and his industrious students at Göteborg (Sweden) produced a hand-parsed section of the Brown Corpus. The 'Gothenburg Corpus', as it has been called, consisted of samples amounting to 128,000 words. In the early 1980s, a team at Nijmegen began the TOSCA system for parsing corpus sentences (see van Halteren and Oostdijk 1993), and the team at Lancaster who had tagged the LOB Corpus attempted the parsing of the same corpus by probabilistic methods (Garside and Leech 1985, Garside *et al.* 1987), although hardware and software limitations prevented the completion of the task.[17] In the later 1980s and early 1990s the building of **treebanks** (i.e. parsed corpora – see Chapter 3) took off as a major activity: it was becoming recognized that syntactically annotated corpora were an important resource for the development of NLP software, for example in the development of robust wide-coverage parsers for such applications as speech recognition and machine-aided translation. The Lancaster/IBM treebank (compiled in

1987–91) comprised about 3 million words (Leech and Garside 1991), and the Penn Treebank initiative (Marcus *et al.* 1993)[18] brought the fruits of this new technology to a wider public of users.

The convenient term 'treebank', commonly used for syntactically annotated corpora, brings to notice the fact that the phrase-structure (PS) tree remains the favoured basic model for corpus parsing. Being a more complex and resource-demanding task than grammatical word-tagging, corpus parsing lags behind grammatical tagging in all respects: it began later, it has been less successful, and has been liable to greater inaccuracy, ambiguity, and incompleteness. Early attempts at parsing have had to make do with simplified constituent-structure models, and hence the term 'skeleton parsing' or 'skeletal parsing' was used to characterize the initial Lancaster/IBM and Penn treebanks (Leech and Garside 1991; Marcus *et al.* 1993).

Corpus parsing is still an evolving technology, but it is evolving at a rapid rate. The current state of the art will be further discussed in Chapter 3, but it is worth noting here that, whereas the first treebank (the Gothenburg Corpus) was entirely annotated by hand, we are now reaching a stage where automatic parsing (without extensive post-editing) is becoming practicable. This trend is perhaps best illustrated by the Constraint Grammar parser of the Helsinki group (Karlsson *et al.* 1995), which, although its output is a partial rather than complete parse, does run relatively satisfactorily over large corpora, and has indeed been used to annotate the Bank of English corpus of more than 300 million words. (The Constraint Grammar formalism is also notable for incorporating a dependency grammar framework, in contrast to the PS models employed for most other treebanks.) Towards the other end of the spectrum of size, but equally significant in its way, is the SUSANNE Corpus which is a manual reworking, in considerable detail, of the Gothenburg Corpus, each decision being justified by a detailed parsing scheme published in book form (Sampson 1995). As with tagging, syntactic annotation has a methodological continuum running from 'entirely automatic' to 'entirely manual'. Somewhere on this continuum is the potentially interactive method employed with increasing success by the Nijmegen group (Aarts *et al.* 1993, van Halteren and Oostdijk 1993), where automatic parsing takes place in an environment allowing or requiring intervention to complete the task of satisfactory parsing.

1.4.4 Other levels

Although most of the effort in corpus annotation so far has gone into work

at the word-class and syntactic levels, other levels of annotation are now beginning to take off: for example, semantic annotation and discoursal annotation. Section 1.5 looks at the different levels of annotation which already exist, summarizing the current state of progress. In Chapters 2–6, these will be explored in greater depth.

1.5 What Levels of Annotation Exist or Can Exist?

Up to now, different levels of annotation have been applied rather patchily, as the list in Box 1.2 (working from the least abstract to the most abstract levels of analysis) indicates. The right-hand column indicates the relevant chapter or section of this book.

As every one of these types of annotation will be discussed and explained in later chapters, the brief illustrations in Box 1.3 are all that are needed at this stage.

Box 1.2 Levels of Corpus Annotation

Linguistic level	Annotations carried out so far	Chapter of this book
Orthographic	This is generally considered part of 'mark up'	(but see §1.5.1)
Phonetic/ phonemic	Widespread in speech science – but typically collected in laboratory situations	(see n.8 this chapter)
Prosodic	Two or three prosodically-annotated corpora are available for widespread use	§6.1
Part of speech (i.e. grammatical tagging)	The most widespread type of corpus annotation, which has been applied to many languages	(Chap. 2)
Syntactic, i.e. (partial) parsing	This is the second most widespread type of corpus annotation, and is rapidly developing	(Chap. 3)
Semantic	Some exists, and more is developing	(Chap. 4)
Discoursal	Little exists – but some is developing	(Chap. 5)
Pragmatic/ Stylistic	(As for discoursal annotation)	(§§6.2–3)

Box 1.3 Brief illustrations of levels of annotation

Example 1a Prosodic annotation, London-Lund Corpus

well ^very nice of you to ((come and)) _spare the
!t\/ime and#
^come and !t\alk# -
^tell me a'bout the – !pr\oblems#
and ^incidentally# .
^I [@:] ^do ^do t\ell me#
^anything you 'want about the :college in "!g\eneral

Example 1b Grammatical tagging from the Penn Treebank, using the Penn Tagset

Origin/NN of/IN state/NN automobile/NN practices/NNS ./.
The/DT practice/NN of/IN state-owned/JJ vehicles/NNS for/IN use/NN of/IN employees/NNS on/IN business/NN dates/VVZ back/RP over/IN forty/CD years/NNS ./.

Example 1c Skeleton parsing (syntactic annotation) from the Spoken English Corpus

[S[N Nemo_NP1 ,_, [N the_AT killer_NN1 whale_NN1 N] ,_, [Fr[N who_PNQS N][V 'd_VHD grown_VVN [J too_RG big_JJ [P for_IF [N his_APP$ pool_NN1 [P on_II [N Clacton_NP1 Pier_NNL1 N]P]N]P]JJ]V]Fr]N] ,_, [V has_VHZ arrived_VVN safely_RR [P at_II [N his_APP$ new_JJ home_NN1 [P in_II [N Windsor_NP1 [safari_NN1 park_NNL1]N]P]N]P]V] ._. S]

Example 1d A type of semantic word-tagging

There_Z5 's_Z5 been_A3+ more_N5++ violence_E3- in_Z5 the_Z5 Basque_Z2 country_M7 in_Z5 northern_M6 Spain_Z2 :_PUNC one_N1 policeman_G2.1/S2m has_Z5 been_Z5 killed_L1- ,_PUNC and_Z5 two_N1 have_Z5 been_Z5 injured_B2- in_Z5 a_Z5 grenade_G3 and_Z5 machine-gun_G3 attack_G3 on_Z5 their_Z8 patrol-car_M3/G2.1 ._PUNC

Example 1e Discoursal Annotation (anaphoric)

(0) The state Supreme Court has refused to release {1[2 Rahway State Prison 2] inmate 1}} (1 James Scott 1) on bail .
(1 The fighter 1) is serving 30-40 years for a 1975 armed robbery conviction . (1 Scott 1) had asked for freedom while <1 he waits for an appeal decision. Meanwhile , [3 <1 his promoter 3] , {{3 Murad Muhammed 3} , said Wednesday <3 he netted only $15,250 for (4 [1 Scott 1] 's nationally televised light heavyweight fight against {5 ranking contender 5}} (5 Yaqui Lopez 5) last Saturday 4) .

1.5.1 Orthographic annotation

Orthographic annotation might seem to be a contradiction in terms – since, as we have seen, orthography represents the text, while annotation interprets the text linguistically. However, up to a point orthographic information can be interpretive, in distinguishing the linguistic functions of various visual devices on paper. Consider different graphological signals for indicating the beginning and the end of a quotation: single quotes, double quotes and change of typesize accompanied by indentation. These are different in form, but alike in function. Conversely, we can also say that the single mark (') is unitary in form, but ambiguous in function: it can signal a single closing quote, or it can represent an apostrophe. Hence, when we read *friends'* out of context, we have to keep both possibilities in mind. Against this background, the TEI Guidelines (Sperberg-McQueen 1991, Sperberg-McQueen and Burnard 1994)[19] allow us to use a pair of symbols &bquo; ('begin quote') and &equo; ('end quote') which signal these orthographic functions irrespective of the typographical device used.[20] This TEI mark-up may be regarded as a kind of orthographic annotation. Other ambiguous orthographic devices which might be annotated to resolve ambiguities are:

1. *Initial* capital letters, which may signal the beginning of a sentence, the beginning of a proper noun, etc.
2. *A period* (.), which may signal either the end of a sentence or an abbreviation
3. *Italics*, which may signal a cited expression, an expression italicized for emphasis, etc.

In some cases, these distinctions have been made in the encoding of the orthographic record of a text, and have hence made a useful contribution to corpus processing and annotation at more abstract levels. For example, in the LOB Corpus the symbol \0 was used to signal a one-word abbreviation (Johansson *et al.* 1978), so that, for example, \0in. as an abbreviation for *inch* could be automatically distinguished from the preposition *in* at the end of a sentence. However, in general such orthographic annotation has not been consistently applied to corpora, so that it would be unwise to rely upon it in designing software for corpus annotation.

1.5.2 Additional types of annotation

In addition to the above levels of annotation, there are some other levels which should be mentioned, although they remain largely undeveloped.

First, there are some levels of linguistic structure or function which could be indicated by annotation, although we know of no generally-available corpora annotated at these levels. For example, at the phonological level, corpora could be annotated by syllable boundaries. At the morphological level, corpora could be annotated by their morphological structure, in terms of prefixes, suffixes and stems. Previous experience suggests that, even if no need for such levels of annotation has yet appeared, such a need is quite likely to arise in the future.

Second, there is the level of **lexeme** annotation or **lemma** annotation (these are alternative names for the same concept). When we have talked of grammatical word tagging previously, we have been assuming that, for example, *eat, eats, ate, eaten* and *eating* receive different tags according to their morphosyntactic function as past tense verb, *-ing* form of the verb, etc. However, another approach to grammatical word tagging would give each of these the same tag, and indicate that they all belong to the lemma EAT ('lemma' being more or less equivalent to the headword of a dictionary entry). In English, lemma annotation may be considered somewhat redundant,[21] since English is a language with simple inflectional morphology. But in more highly-inflected languages, such as Russian or Spanish, there is a relatively large number of word-forms per lemma, so that lemma annotation may have a valuable contribution to make to information extraction – for example, for the improvement of dictionaries or computer lexicons of the language.

Third, there is a kind of annotation which does not depend on the simple recognition of different levels of linguistic function, but is more closely geared to applications. Thus, there have recently come into being a number of **learner corpora** of English, representing the language of those learning English as a second or foreign language (e.g. Granger 1993). The function of such corpora is to advance our knowledge of how languages are learned as a second language: for example, to what extent does the English of non-native speakers reflect the influence of their native tongue? For this kind of investigation, it is very useful to annotate the corpus with classes of errors, or features of non-native language behaviour. Such 'error tags' make use of grammatical and lexical classifications, for example, but also take into account the relation between the non-native and corresponding native phenomena.

'Error tagging' of learner corpora is just one example of application-oriented annotations, and there may be many more. This is sufficient to indicate that annotation is an open-ended area of research, which is very much under development. While in the next five chapters we review levels of annotation which already exist, it cannot be doubted that new kinds of annotation will arise in the future.

Notes

1. The corpus was originally more verbosely labelled 'a standard sample of present-day edited American English for use with digital computers' (see Francis and Kučera 1964).
2. As the BNC will be a focus of discussion in a number of chapters, it will be useful here to add some details of its compilation and composition. The BNC is a corpus of 100 million words, containing texts taken from sources such as newspapers, books, magazines, and transcribed conversations, lectures, meetings and interviews (Burnard 1995). The corpus is also annotated, in that individual words have been tagged to show part-of-speech (POS) information. The whole of the BNC has been tagged using the relatively small C5 tagset (see Appendix III) while 2 million words of the corpus, known as the sampler corpus, have been tagged using an expanded version of the tagset, known as the C7 tagset (consisting of 146 POS tags). The corpus was built by a team consisting of Oxford University Press, Longman Group Ltd., Chambers Harrap, The British Library and the Universities of Oxford (Oxford University Computing Services) and Lancaster (UCREL). Further details of the BNC are provided by Burnard (1995); a broad survey is given by Leech (1994).
3. Information on the Bank of English can be accessed on the World-Wide Web at http://titania.cobuild.collins.co.uk/boe_info.html.
4. In fact, this is not quite true. Experiments have been carried out to induce linguistic word classes from a corpus purely automatically, on a distributional basis (see, e.g., Atwell and Elliott 1987), making no use of humanly-devised categories. Such classes sometimes have an uncanny resemblance to categories used in grammatical or semantic tagging (e.g. prepositions, modal auxiliaries, nouns for months). Whether a labelling of corpus words according to these induced categories would be considered a kind of linguistic annotation is a matter of terminological definition.
5. Much of this encoding is purely conventional: the ASCII code is the encoding system generally used for converting the symbols on the terminal or typewriter keyboard to binary electronic form.
6. There is, of course, more than this to the issue of 'what is the purely orthographic record of a text'. Some compilers of corpora have been content with the 'plain ASCII text', without mark-up indicating such linguistically-relevant details as headings and highlighted expressions. Going to the other extreme, others will take the view that any diagrams, photographs, etc., accompanying a written text are as much a part of it as the words themselves. However, these are still issues of what comprises the representation of a text: they do not trespass into the 'metalinguistic' territory of corpus annotation.

   Another kind of information provided in a corpus may be considered distinct from both the text itself and the annotation of the text. This is **header information** (so-called because it tends to be provided in headers, or headings, at the beginning of a text or corpus). This gives information of various kinds about the 'documents' or texts which comprise a corpus, as well as

about the corpus in its entirety. Such information may include bibliographical details of a written text, and parallel information about a spoken discourse (regarding identity and background of speakers, the provenance of the transcription, etc.). It may also provide a classification of the 'document' in terms of the text typology used in designing the corpus, hence giving information of an interpretative, linguistic nature – for example, indicating something of the style of language found in the 'document'.

7. For spoken discourse, papers by Ochs (1979) and Cook (1995) deal with issues connected with the non-objectivity of theory. Their papers are provokingly, though aptly, named 'Transcription as theory' and 'Transcribing the untranscribable'.
8. However, we do not attempt to cover in this book the subject of **speech corpora**, by which is generally meant recorded and annotated speech data collected under 'laboratory' conditions, i.e. conditions not comparable to those of authentic spoken discourse. The immense amount of recent work on speech corpora, including annotation, can be seen by consulting the EAGLES World Wide Web site http://coral.lili.uni-bielefeld.de/~gibbon/EAGLES/. A further source of information is the handbook of the EAGLES Spoken Language Standards and Resources Group due for publication in 1997. See also Section 6.1.2 (1).
9. It is notable that for official purposes in society at large, such as the transcription of court proceedings, a verbatim transcript corresponding faithfully to the words spoken, in their right order, is considered to be a faithful record of what was said.
10. Here we are making an assumption that the **user** of an annotated corpus is not the same as the **annotator**. It is possible, of course, that some annotations are done by researchers purely for their own use, with no intention of distributing their annotations to others. However, as far as this book is concerned, the reason why annotation is worth studying in depth is that in natural language processing (NLP) it is increasingly becoming important to re-use the resources compiled or devised by others (see Section 1.2.2). For us, then, the users of a corpus comprise a potentially large group, typically distributed across the world, and engaging in many different kinds of research and development activity.
11. Sampson (1995) and Johansson (1986) are two detailed examples of what an annotation scheme should attempt to do. An annotation scheme should include: (i) a list of the annotative symbols used, (ii) their definitions, and (iii) the rules or guidelines that have been used in their application. Another way in which an annotation scheme can explicate the nature of the annotations is to cross-refer (for instance) to a lexicon or a grammar or a 'reference corpus' which exemplifies the various descriptive decisions made by the annotators.
12. In the interests of **re-usability** (see Section 1.2.2).
13. 'LOB' Corpus is an abbreviation for the Lancaster-Oslo/Bergen Corpus, compiled at the three universities mentioned in its name during 1970–78 (see Johansson *et al.* 1978).

14. Early work was undertaken on the tagging of Swedish at Lund and of Dutch at Nijmegen.
15. The Survey of English Usage project was announced and described in Quirk (1960). The majority of the spoken materials of the LLC have also been published in book form (Svartvik and Quirk 1980).
16. See Knowles (1993). The SEC was later reworked as a CD-ROM where all levels of annotation were combined in a single database, and were cross-registered to the digital soundtrack and the F_0 waveform.
17. The parsing of about 144,000 words of the 1-million-word LOB Corpus was eventually completed, and made available via the Norwegian Computing Centre for the Humanities, under the title of the 'Lancaster Parsed Corpus'.
18. A subset of the Penn Treebank is available to researchers for non-commercial purposes, on payment of a license fee, from the Linguistic Data Consortium (LDC). For further details, see the relevant items on the LDC's World Wide Web site: http://www.ldc.upenn.edu/.
19. 'TEI' stands for the Text Encoding Initiative, an international initiative to set up a flexible standard for the encoding or mark-up of texts for electronic interchange. For most purposes, we may see the TEI as systemizing the representation of raw text, rather than as being concerned with annotation practices. However, there is a sense in which TEI mark-up is an aspect of annotation practices: it lays down guidelines for the representation of annotations. Just as the raw corpus needs to be represented electronically, so the annotations need to be represented electronically. And it is this aspect of annotation practices (and not, say, the choice of categories) which comes within the purview of the TEI (see further Section 2.4).
20. It needs to be said that SGML, the language which TEI uses, attempts to mark up an original text by **function** rather than **realization**. Thus a word to be emphasized is to be marked as such, rather than as **italic**, and the realization of emphasis (as well as, say, foreign words) by italics would be specified independently.
21. However, lemma annotation has been undertaken by Fligelstone (1995) for a corpus of English newspapers and by Sampson (1995) for the SUSANNE Corpus.

2

Grammatical Tagging

GEOFFREY LEECH

In this and the following chapters, we take a closer look at the linguistic nature of grammatical annotation, comprising **grammatical word tagging** and **syntactic annotation**. (Chapters 7 to 11 return to these levels of annotation, and examine the software designed to help carry out these tasks.) The reason for considering grammatical tagging and syntactic annotation together is obvious: they both, in a general sense, specify the grammatical characteristics of a text. In fact, there is a strong argument that these are not really distinct levels at all: grammatical tagging is merely a specification of the leaves (or pre-terminal nodes) of the phrase-structure (PS) tree which is a favoured model for syntactic annotation (Figure 2.1). No doubt, in a decade or two, there will be adequate working corpus parsers which will carry out both levels of annotation as a single integrated process.

Nevertheless, at the present time, grammatical tagging and syntactic annotation are often considered separate tasks, the former being preliminary to the latter.[1] Considering the difficulty of the task of parsing unrestricted text,[2] it is in any case a useful expedient, in the present state of the art, to divide the work of parsing into two manageable, though individually quite complex, tasks, rather than to attempt the more challenging task of integrated corpus parsing. Correspondingly, it is useful to treat the two

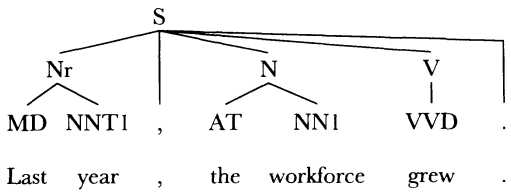


Figure 2.1 Simple tree diagram showing grammatical tags as pre-terminal nodes of a PS tree

tasks in separate chapters, because the techniques for tagging and parsing have evolved to a considerable extent along separate paths.

2.1 A Tagging System

When beginning the task of grammatical tagging, the annotator has to consider at least three questions:

1. How to divide the text into individual **word tokens** (or words)?
2. How to choose a **tagset** (or a set of word categories to be applied to the word tokens)?
3. How to choose which **tag** is to be applied to which **word token**?

These are all basically linguistic questions. The answers to them will add up to a linguistic specification of how the tagging is to be done, which may be called the **tagging system**. However, these answers are also likely to be heavily influenced by non-linguistic issues, such as:

(a) Is the task going to be performed automatically or manually (or, more likely, automatically with manual post-editing)?
(b) If there is going to be automatic tagging, what are the techniques and capabilities of the tagging software?
(c) What human and hardware resources are available for the task?
(d) How quick, how accurate and how consistent does the resultant tagging need to be?

However much we may want to ignore these practical and technical questions, the answers to them are likely to prevent us from achieving a linguistically optimal result. In fact, it is rather artificial to separate (for example) the choice of a tagset from the development of tagging software, which may well be proceeding in parallel with the development of the entire tagging system.

When undertaking a corpus tagging project, annotators often begin with a rough outline of a tagging system, and this provides a set of preliminary guidelines which will undergo revision and refinement as the project progresses. For both the annotators and the users of a corpus, answers to questions (1)–(3) above are best handled by a **Tagging Manual** completed at the end of the project, and retrospectively taking account of any changes introduced in the course of tagging. This documentation should be available to users, and should ideally be distributed with the corpus itself. Two good examples of a tagging manual are those for the LOB Corpus (Johansson 1986) and for the SUSANNE Corpus (Sampson 1995, Ch. 3).

2.2 Tokenization: multiwords, merged words and 'phantom words'

The first issue in (1)–(3) above, how to segment the text into word tokens,[3] appears at first glance to be trivial. A written text normally reaches us with word tokens clearly demarcated by a preceding and following space character or new-line character, which may in addition be accompanied by punctuation marks. For modern languages with alphabetic writing systems, the **orthographic word** is automatically identifiable in the representation of the text itself.[4] However, an orthographic word (recognizable on the page by the white space preceding and following – but not interrupting it) is not necessarily the same as the word as a morphosyntactic unit: that is, the word token that we need to identify for the purposes of grammatical tagging.

There are three main kinds of deviation from the one-to-one relation between orthographic and morphosyntactic word tokens. These are fortunately relatively infrequent, so that the one-to-one correspondence between an orthographic token and a morphosyntactic token can be considered the default case that applies in the absence of special conditions. The three exceptional conditions are as follows:[5]

1. **Multiwords**: *more than one orthographic word corresponds to one morphosyntactic word.*
   For example the sequence *in spite of* consists of three orthographic words, but will more usefully be tagged as a single preposition: i.e., as a single morphosyntactic unit. One convenient way to annotate multiwords is to label each orthographic word with the same tag, followed immediately by two digits *xy*, the *x* indicating the number of tokens in the multiword, and the *y* indicating the *y*th token. Thus 21 at the end of a tag means 'the first part of a two-part multiword', and 22 means 'the second part of a two-part multiword'. The individual tags with the appended digits are referred to in this book as **ditto tags**.[6] For example:

provided_SCONJ21 that_SCONJ22	(Multiword subordinating conjunction)
in_PREP31 spite_PREP32 of_PREP33	(Multiword preposition)

   Two questions which should be answered in a Tagging Manual are:

   a. Are discontinuous multiwords allowed (an example might be phrasal verbs in English or separable verbs in German)?
   b. Which sequences are classified as multiwords, and which are not?

(For example, *provided that* might be treated as a multiword in one tagging system, but not in another.)

2. **Mergers**: *one orthographic word corresponds to more than one morphosyntactic word.*
   Most of these cases involve **clitic** forms, that is word forms which are phonologically reduced and which show up in writing as being orthographically attached to another word. **Proclitic** forms are those which are attached to the front of another word (as in French *Je t'aime*, where *t'*, an elided form of the pronoun *te*, is attached to the front of the verb). **Enclitic** forms are those which are attached to the end of another word (as in English *hasn't*, where the negative particle *not* is reduced to *n't* and added to the end of the verb). Other cases of merger are purely orthographic, not involving reduction or the use of the apostrophe (e.g. Italian *vendetelo* 'Sell it' consists of an imperative verb followed by an appended pronoun *lo*). Still other cases are problematic because the merged form cannot be easily divided into separately spelt words. An example is the French form *du* (= *de* 'of' + *le* 'the'), and the English negative auxiliary verb *shan't* (= shall not). There is no generally accepted way to represent orthographically merged forms, but one solution – although care should be taken to avoid ambiguity in using it – is simply to run the words with their tags together, without intervening spaces:

   vendete_VERBlo_PRON t'_PRONaime_VERB sha_VAUXn't_NEG

   One drawback of this method, evident from the last example, is that it leads to the creation of artificial 'phantom words' like *sha*, not to be found in any English dictionary. The colloquial spelling *dunno* (a merger of three word tokens *do* + *not* + *know*) is even more problematic in this respect.
3. **Compounds**: *depending on the analysis,* ***one*** *or* ***more than one*** *orthographic word corresponds to* ***one*** *or* ***more than one*** *morphosyntactic word.*
   This is a very open-ended category. Basically, a compound may be defined as a word which has other words as its components. But in practice, it is difficult to draw either an upper bound or a lower bound for the identification of compounds. With a word like *rainbow*, the word is historically derived from two component words *rain* and *bow*, but in present-day English the compound is virtually fossilized. To all intents and purposes, *rainbow* is a single noun in the English lexicon, and only marginally to be considered a compound. With a sequence like *word class*, on the other hand, *word* and *class* are spelt as two separate nouns, although some might prefer to spell the sequence with a hyphen (*word-class*), to signal their incipient compound status. So here, the sequence

is somewhere in the 'grey area' between being analysed as a single compound and being analysed as a sequence of two stand-alone nouns. In fact, the orthographic signalling of compounding is highly variable in English, and the same expression (e.g. *eye* + strain) may be written *eye strain, eye-strain*, or *eyestrain* according to stylistic taste. In these circumstances, it seems linguistically safest to represent word boundaries on two different levels, which are here represented by bracketing:

(a) [[eye_NOUN][strain_NOUN]_NOUN]
(one orthographic word without hyphen)
(b) [[eye_NOUN]-[strain_NOUN]_NOUN]
(one orthographic word with hyphen)
(c) [[eye_NOUN] [strain_NOUN] _NOUN]
(two orthographic words separated by spaces)

(a)–(b) are then three ways of representing *eye* + *strain* as the same compound, according to whether the spelling *eyestrain, eye-strain* or *eye strain* occurs in the original. This is the 'safe' course, because, whichever orthographic variant occurs, the fact that the sequence *eye* + *strain* is recognizable as a single compound noun is retrievable in each case. In a way, it shows that the distinction between syntactic annotation and grammatical tagging, with which this chapter began, is not discrete: the analysis of word tokens may also involve 'parsing' into tree-diagram-like structures.

However, in practice the tagging of corpora has not meddled with compounding in such depth. The usual practice has been to let the orthography determine whether a compound is tagged as one word or as two. In that case, *tagset* will be tagged 'tagset_NOUN' and *tag set* will be tagged 'tag_NOUN set_NOUN'. The third alternative, *tag-set*, will receive varying treatment according to whether the hyphen is regarded as word-internal or word-external punctuation. Probably, the usual choice is to treat the hyphen as word-internal, so that the tagging will be 'tag-set_NOUN'.

The 'cheap and cheerful' way of dealing with compounds just illustrated bypasses the whole aggravating business of deciding when an expression is a compound or not, treating the default correspondence of orthographic and morphosyntactic words as an expedient way of forcing a decision. However, the drawback is that if a user wants to use the tags as a means of extracting compound expressions (a very pressing need in terminology extraction, for example), three different searches may have to be made, for *tagset, tag set* and *tag-set*.

A further drawback of this approach is that a second kind of 'phantom word' phenomenon may occasionally occur, exemplified by the

'word' *York-San* in *New York-San Francisco flights*, or the 'word' *post-Cold* in *post-Cold War attitudes*. As anyone who knows English can tell, linguistically these do not make sensible words, and cannot reasonably be assigned a word tag. But they result from the automatic application of the default rule 'orthographic word corresponds to morphosyntactic word'. The more complicated analysis, which would permit a logical solution, would be to assign names such as *New York* a single 'compound' tag (treating them as multiwords), while treating the hyphen as a word-internal link. The overall sequence *New York-San Francisco* would then be considered a larger compound, as indicated by the following bracketing (where each bracket represents a word, to be assigned its own tag):

(a) [[New York]-[San Francisco]]
(b) [[post]-[Cold War]]

Fortunately, such cases are rare, but they illustrate the difficulties that tokenization occasionally runs into.

2.3 Tagsets

A tagset is simply a list of tags used for a given task of grammatical tagging. Tags represent word-categories, but there is clearly room for disagreement about what word-categories are useful or linguistically applicable. There is also room for interference from practical constraints – such as the need for speed and accuracy in automatic tagging. An 'armchair linguist' might devise a tagset based on sound linguistic principles, only to discover that a particular tag is incapable of being automatically assigned with any degree of accuracy. If that 'armchair linguist' had to face the consequences of the (linguistically irreproachable) decision to recognize such a category, for example in manually correcting a few thousand examples, he or she might find the expedient solution more attractive than the linguistically preferable one!

One example of this from our own experience at Lancaster is having separate tags for the present subjunctive (***come*** *what may*) and the imperative form (***come*** *here!*) in English. Although these categories make good grammatical sense (and correspond to clearly identifiable categories in related languages), in practice they are indistinguishable from the present tense plural indicative form (*They* ***come*** *every spring*). In the current state of the art of grammatical tagging, it is difficult to distinguish subjunctives and imperatives from other base verb forms without a substantial proportion of errors. The solution we arrived at was to merge the three categories

'indicative', 'imperative' and 'subjunctive' into a single category 'finite base form' (the **base form** being the form of the verb which has no inflectional ending or vowel mutation), distinguishing this from the non-finite base form (viz. the infinitive, as in *Would you like to* ***come****?*). A further stage of simplification is to ignore the infinitive-finite distinction, and simply assign one tag to all the base forms of English lexical verbs. This is what was done in the tagging of the Brown Corpus and of the LOB Corpus, and also in many other tagging projects applied to the English language.

As this example suggests, there normally has to be a trade-off between what is linguistically most desirable and computationally feasible. Where the bargain is struck between these two factors depends on the circumstances of individual projects. In general, however, automatic taggers take account only of the immediate local context of a word. If a grammatical distinction is difficult to make successfully using local context, it will probably be abandoned in the tagging system.

2.3.1 Tags and labels

It is useful to make a distinction here between **tag** and **label**. A tag is a word-class embodied in an annotative device associated with a word in the text. But there can be many ways of encoding that category in terms of alphanumeric characters. The part of speech **preposition** may be encoded as Preposition, prep, or (less mnemonically, in the Brown and LOB Corpus tagsets) as IN. In a more specific category, such as **singular proper noun**, there are three pieces of information to convey (singular vs. plural, proper vs. common, and noun vs. verb, pronoun, etc.), and these may be reflected in the label, which might be Noun: prop: sing, or N-p-sg, or (following the C7 tagset – see Appendix III) NP1. There is again a trade-off here – between ease of human processing and ease of machine processing; also between conciseness and perspicuity. Three criteria to bear in mind when choosing labels for tags are:

1. **Conciseness** Brief labels are often more convenient to use than verbose, lengthy ones.
2. **Perspicuity** Labels which can easily be interpreted are more user-friendly than labels which cannot. This means that tags should be, where possible, easily remembered: Preposition, by this criterion, is better than IN.
3. **Analysability** Labels which are decomposable into their logical parts are better (particularly for machine processing, but also for human understanding) that those which are not. For example, NP1, in the BNC tagset (see below) can be decomposed into:

N= noun (vs. V = verb, P = pronoun, etc.)
P = proper [noun] (vs. N = common noun)
1 = singular (vs. 2 = plural)

One major advantage of analysability is that searches of the corpus (and other automatic processing tasks) can be carried out at varying levels of granularity. For example, the symbol N* can represent all nouns (* being a **wildcard symbol**, matching any string of characters, including the zero string). The symbol N*1 can, similarly, represent all singular nouns, and the symbol NP* all proper nouns, whether singular, plural or neutral for number. It must be admitted, however, that NP1 scores relatively low on the perspicuity scale: many people prefer to use more verbose but recognizable labels such as Noun: prop: sing.

One can, however, make too much of these criteria. So long as **disambiguity** (4) is preserved, it is possible, and trivial, to convert one label into another automatically. The tags of a corpus, then, may be given labels which vary according to the purpose for which they are to be used. Internally, for machine processing, priority is likely to be given to criteria (1), (3) and (4); externally, for human friendliness, priority is likely to be given to criterion (2).

While we are considering presentation, it should be noted that a tagged corpus can be presented either in a horizontal or in a vertical format. In Boxes 2.1 and 2.2, the same corpus sentence is shown in each format:

Box 2.1 Horizontal Format

```
Oh_UH ,_, and_CC he_PPHS1 did_VDD pass_VV0 his_APP$ exams_NN2 ._.
```

Box 2.2 Vertical Format

```
SK01 271  Oh     UH     (interjection)
SK01 272  ,      ,      (comma)
SK01 273  and    CC     (coordinating conjunction)
SK01 274  he     PPHS1  (3rd pers. pronoun, sing. nom)
SK01 275  did    VDD    (past tense of the verb do)
SK01 276  pass   VV0    (base form of lexical verb)
SK01 277  his    APP$   (possessive determiner)
SK01 278  exams  NN2    (plural common noun)
SK01 279  .      .      (period)
```

Again, a corpus or a text may be trivially converted from one format to the other. But the choice of format may have consequences for the choice of labels: for example, verbose labels are more conveniently handled in the vertical format than in the horizontal one.

2.3.2 *Logical tagsets*

The idea of a **logical tagset** is that the relations between the word categories symbolized by tags should be representable as a hierarchical tree (not a PS tree, but a tree of features and attributes), with attributes being inherited from one level of the tree to another.[7] Let us take the example of the C7 tagset, applied to a part of the BNC by the Lancaster team: at the most general level, the tree distinguishes part-of-speech categories: N=noun, V=verb, J=adjective, R=adverb, P=pronoun, D=determiner, A=article, C=conjunction, M= numeral, I=preposition, etc. Beneath each of these major parts of speech, there are subordinate attributes such as P=proper and N=common for nouns, or P=personal and N=indefinite for pronouns. (Note that the meanings of these symbols are context-dependent, such that P following an N means 'proper', whereas P following another P means 'personal'. It should also be noted that the same alpha-numeric symbol, in a particular position in the sequence of symbols in the label, may have the same meaning across different branches of the tree: for example, Q, as the third character, signifies a *wh*-word; 1 and 2, as the final character, signify singular and plural.) Figure 2.2 overleaf is a summary representation of part of the C7 tagset as a hierarchical, logical tagset. At the right-hand side of each line, a tag label (such as NN2) is given, together with a brief definition of the tag, and (in brackets) one or two examples. The whole of the C7 tagset is listed in Appendix III.

2.3.3 *The size and composition of tagsets*

The size of a tagset is less important than it might seem. It is relatively easy to increase or decrease the size of a tagset, according to the emphasis that a particular project has. The 'core' of a tagset will tend to be major word classes with their principal sub-classes, as illustrated above. But perhaps of more note are the peripheral elements which need to be marked in a corpus and which tend to be ignored unless, as a corpus analyst, we are doggedly trying to assign a meaningful tag to each and every word token in a text. For a written corpus, certain categories of **WIC** (word-initial capital) words are easy to recognize, and can be semantically and syntactically significant in their own right, even though they do not feature in traditional morphosyntax. Examples are month nouns (*January*), day

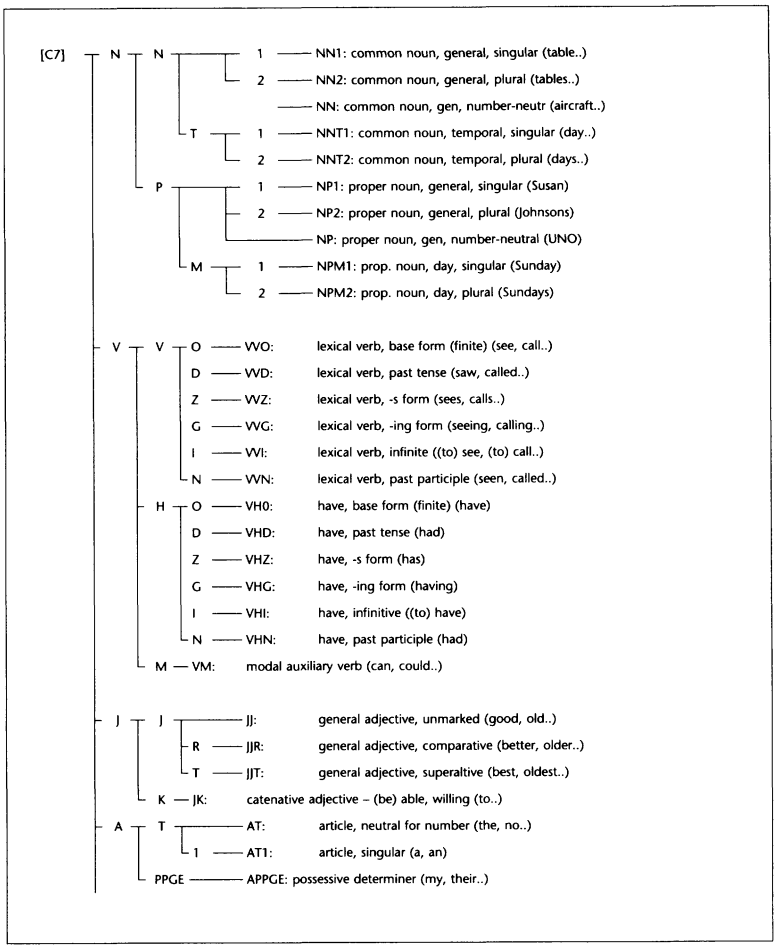


Figure 2.2 Part of the C7 tagset presented as a logical tagset

nouns (*Tuesday*), and adjectives and common nouns derived from proper nouns (*French, Frenchman*). In spoken corpora, on the other hand, it would be more useful to distinguish certain types of discourse marker (*well*) or hesitation marker (*erm, er*), rather than to lump these together with interjections (*oh, ah*). Some tags represent singleton word classes with only one member: in English, existential *there*, the negative particle *not* and the infinitive marker *to* are often each given a unique tag because of their unique syntactic behaviour. There remains, when these have been dealt with, a residuum of phenomena such as formulae (*P23, C: \WINDOWS*, etc.) and

foreign words (*Timeo Danaos et dona ferentes*), and punctuation marks (, . ! ...) which, although they are not words, are conveniently treated as words for the purposes of grammatical tagging.[8]

Given these various components of a tagset, tagsets for English tend to vary between 30 and 200 members. Examples are given in Table 2.1. The number increases greatly for tagsets including subcategorization features for valency, such as 'transitive' and 'intransitive' for verbs.

Table 2.1 Some tagsets for English

TOSCA tagset (van Halteren and Oostdijk 1993)	32
Penn tagset (Marcus and Santorini 1992)	36
BNC C5 tagset	61
Brown tagset (Greene and Rubin 1971, Francis 1980)	77
LOB tagset (Garside *et al.* 1987)	132
London-Lund Corpus tagset (Svartvik and Eeg-Olofsson 1982)	197
TOSCA-ICE tagset (van Halteren forthcoming)	270

For one of the Spanish tagsets discussed in Section 10.4.1, the number of tags is as high as 475. This is mainly because Spanish, in comparison with English, has many different verb inflections: there is a general tendency for tagsets to increase in size proportionate to the richness of a language's inflectional morphology.

Sometimes there is a conflict between linguistic (or 'external') reasons and computational (or 'internal') reasons for determining the composition of a tagset. The linguistic quality of a tagset (e.g. the extent to which it allows retrieval of all important grammatical distinctions in the language) is 'external' in the sense that it concerns the user's requirements. The computational tractability of a tagset (e.g. the extent to which a particular tag is useful in aiding the disambiguation process, and increasing the accuracy of tagging) is in contrast 'internal'. Most tagsets show some signs of the 'internal' criteria impinging on the 'external' (cf. the discussion above of the low tractability of the subjunctive category, given the ambiguity of verb base forms in English). On the other hand, tags with a high tractability are those which are unambiguously assigned to particular (orthographic) words, and which have a high value in disambiguating neighbouring words. An example is the tag (AT in the C7 tagset) for the definite article in English. As explained in note 8, one of the chief reasons why punctuation marks are frequently included in tagsets is that they have an important discriminatory value in defining the contexts in which other tags are likely to occur.

2.4 Encoding of Tags

As already discussed in Section 2.1, issues of tokenization raise a number of problems for the way we encode tags and their associated word tokens. For English, a certain degree of common practice has grown up through the precedent set by the tagging of the Brown Corpus which has been a model imitated to a greater or lesser extent by other tagging projects (e.g. the tagged LOB Corpus, the Penn Treebank, and the SUSANNE Corpus – see Sampson 1987c for a comparison of these tagsets). On the other hand, another somewhat more recent influence has been the Text Encoding Initiative (TEI) (Sperberg-McQueen and Burnard 1994, Burnard 1995), a large-scale movement towards achieving an acceptable standard in the encoding of electronic textual material on computer, particularly for purposes of data interchange, based on the mark-up system known as SGML (Standard Generalized Mark-up Language). The TEI guidelines, still under evaluation and development, cover a very broad spectrum of textual phenomena in a wide range of languages, including the mark-up of written texts, and transcription features of spoken discourse. More recently, moves have been made to apply the TEI guidelines to linguistic annotations, particularly (in the first instance) to grammatical tagging. The recent provisional recommendations for grammatical tagging representation in the European Union's EAGLES initiative (see Ide and Véronis 1995) include the suggestion that the raw text and the annotations should be stored in different files, with cross-referencing between them. This would have one advantage: the 'phantom word' problem for expressions like *dunno* would disappear, but it may cause inconvenience of a different kind.

The only large-scale corpus project so far to work up an extensive and detailed TEI encoding, including an encoding of grammatical tags, is the British National Corpus project, where grammatical tags are marked up as indicated in Box 2.3.

Box 2.3

```
<w AV0>Even <w AT0>the <w AJ0>old <w NN2>women
<w VVB>manage <w AT0>a <w AJ0>slow <w UNC>Buenas<c PUN>,
<w AV0>just <w CJS>as <w PNP>they<w VBB>'re <w VVG>passing
<w PNP>you<c PUN>.</PUN>
```

This example employs the C5 tagset (see Appendix III). Grammatical tags are represented as SGML tags with the 'w' signifying 'word'. The SGML tag

precedes the word to which it applies. (A closing tag containing '/' may also be added: but this is normally omitted where its presence would be redundant.)

Although these labels are less transparent to read on paper or on the computer screen, they do have the advantage of conforming to an international standard, and there is no reason why, by a trivial conversion program, they should not be replaced by a more user-friendly format (e.g. the use of the underline as attachment symbol) for local 'in house' use.

Another advantage of these SGML-conformant set-ups is that they can easily be elaborated to deal with the 'special cases' of multiwords, mergers and 'phantom words' (see Section 2.2). Examples are as follows:

- Multiwords: <w PRP>in lieu of <w NN1>payment
  (Ditto tags are not used here, it being inferred that the tag <w PRP> applies to all three orthographic words following.)
- Mergers: <w PNP>they<w VBB>'re <w VVG>passing
  (Here the merger of *they're* is shown by the simple device of omitting a space between two tags.)
- 'Phantom words': <w AJ0><w PRP>post-</w PRP><w AJ0>Cold</w AJ0> <w NN1>war </w NN1></w AJ0>
  (In effect, the 'phantom word' phenomenon disappears, the assignment of <w > tags being disassociated from the orthographic spaces in the text. The compound structure of *post-Cold War* is shown by the embedding of the three 'constituent words' inside the compound word (in fact, an adjective) comprising the whole sequence. The mark-up tag containing an oblique (e.g. </w XXX> signals the end of the word whose beginning is signalled by <w XXX>).)

2.5 Tagging Schemes: assigning tags to words

As was noted in Section 2.1, an annotation scheme is more than just a list of the symbols used, and a definition of their meaning. A tagging scheme (as we may call an annotation scheme for grammatical tagging) should ideally specify how decisions are made about how to assign tags to words. A lexicon, to some extent, will give relevant information: it will say which tags are assignable to which words. But, where more than one tag can be assigned to the same word, the issue becomes a matter of defining the contextual conditions of choosing this tag or that for a particular word-token. More or less detailed tagging manuals have been produced for various tagged corpora: for example, Johansson (1986) for the LOB Corpus, Santorini (1990) for the Penn Treebank, and Sampson (1995, Ch. 3) for

the SUSANNE Corpus. However, there is virtually no limit to the detail that might be provided, if one were to specify completely and explicitly the tagging decisions to be made in all possible contexts: the notion of a 'complete' tagging scheme is beyond the horizons of contemplation.

For English, certain 'grey areas' of unclarity between the use of one tag and another are notorious. For example, nouns, when they premodify other nouns, resemble adjectives. Should substance words like *gold* in *gold watch* or *plastic* in *plastic bottle* be tagged as nouns or adjectives? Similarly, the boundary between proper nouns and common nouns is particularly uncertain, since common nouns are often assigned word- or sentence-initial capitals. Which of these should be tagged as proper nouns: *[the] Pope, Auntie, Gold* (in *Gold Coast*), *IBM, de* (in *de Gaulle*), *Times* and *Square* (in *Times Square*), *Fifth* (in *Fifth Avenue*), *T* and *S* (in *T. S. Eliot*), *Microsoft* and *Word* (in *Microsoft Word*)? One solution would be to treat two-word and other complex names as 'multiwords', in the sense that a single tag is applied to a whole name such as *Gold Coast* or *Times Square*. As was emphasised in Section 1.3 (4), there is no 'God's truth' in annotation practices: but a tagging scheme, if it is to form the basis for consistent tagging, should give an answer to such questions as these, even if the answer has to be a fairly arbitrary one.

2.6 Conclusion

In this chapter, we have examined the grammatical tagging of English text from various linguistic points of view, so that what initially may have appeared to be a rather dull mechanical labelling task turns out not to be so straightforward after all, and not by any means lacking in challenge and interest. Later chapters, Chapters 7–10, return to grammatical tagging, focusing on automatic tagging, and the development of tagging software. In a later chapter still, Chapter 17, the problem of consistency of annotation is investigated, with focus again on grammatical tagging as the most fully researched annotation level. Meanwhile, we move on in Chapter 3 to the related topic of syntactic annotation.

Notes

1. A different view is taken by Karlsson *et al.* (1995), whose Constraint Grammar parser undertakes tagging and a partial parsing as part of the same task.
2. The notion of unrestricted text (i.e. textual material which has not been artificially 'censored' or selected in advance) is important for NLP, where until recently parsers and other linguistic analysis software were designed and

tested using artificially simplified sets of data, representing a convenient subsample of the language, rather than being capable of analysing any phenomenon that might be found in any text in the language. One important advantage of a corpus-based approach to software development is that a corpus, being an authentic and uncensored sample of the language in use, is a realistically challenging test-bed against which to measure the performance of a parser. Parsers which measure up to the task of analysing any authentic sentence in the language are labelled **robust**. (See Briscoe 1994.)

3. Traditionally in linguistics a distinction is made between **word tokens** (each instance of a word in a text counts as one token) and **word types** (each word as listed in a dictionary, of which tokens are instances, is a word type). **Tokenization,** as the segmentation of a text into words, is therefore the procedure of identifying word tokens in this traditional sense. Sometimes the type-token ambiguity of the word 'word' needs to be resolved by the use of 'word token' and 'word type' as separate terms. In this book, however, we will avoid this cumbersome practice by simply using the term 'word' where the interpretation as type or token is clear from context. Incidentally, the 'type' / 'token' distinction applies also to other kinds of linguistic units, such as phrases, sentences, word tags and lemmas.
4. For other languages, such as Chinese, the identification of word tokens may be a difficult task requiring quite complex computer processing (see Wu and Tseng 1993 on the word tokenization of Chinese).
5. In what follows we use, for clarity's sake, a provisional way of encoding word tokens and their tags under various conditions. Later, in Section 2.4, we look at a more systematic way of encoding them.
6. This assumes, of course, that multiwords contain less than 10 orthographic words. In practice, multiwords rarely extend beyond 3 or 4 orthographic words.
7. The concept of a logical tagset has been developed by the Stuttgart group led by Ulrich Heid. See Teufel (1995).
8. Punctuation marks such as commas and full-stops are considered useful for automatic annotation. For example, in automatic tagging, punctuation marks are often good predictors of preceding and following word-classes. Similarly, in parsing (as demonstrated by Briscoe and Carroll 1996: 146–7), inclusion of punctuation marks as terminal nodes of a parsetree improves the coverage and performance of a corpus parser. (The use of punctuation in parsetrees is illustrated in many of the examples in Chapter 3.)

3

Syntactic Annotation: Treebanks

GEOFFREY LEECH and ELIZABETH EYES

As we have seen, syntactic annotation is the practice of adding syntactic information to a corpus, by incorporating into the text indicators of syntactic structure. For the purposes of this chapter, let us assume that a corpus has already been grammatically tagged (see the beginning of Chapter 2): the syntactic annotation is then the subsequent stage of assigning to corpus sentences such syntactic analyses as labelled bracketing, dependency relations between words, or functional labelling of elements such as subjects and objects. As a baseline strategy, let us consider the task of assigning a straightforward, fairly simple **phrase-structure** (PS)[1] analysis (often called a labelled bracketing) to every sentence in a corpus.

3.1 Why Annotate? The Uses of Parsed Corpora

The first question to ask is: why undertake syntactic annotation of a corpus? As with grammatical tagging, there are many different reasons why an annotated corpus may be more valuable to the user than a raw corpus. In general, the goals which are served by a grammatically tagged corpus are also served, *a fortiori*, by a parsed corpus. Here are two further major reasons for syntactic annotation:

1. **Developing Parsers** Perhaps the most important reason for annotating a corpus is for the training and testing of parsers. A parser is a key piece of software for most applications in NLP: many people believe that to get at the precise meaning of a text, we need first to decode its syntax. If sentences cannot be analysed into their significant components and relations, little progress can be made (it is argued) in the task of using computers to make sense of human language for such purposes as speech recognition and machine translation. Yet over more than twenty years of research, computer grammars and the parsers

implementing them have still lacked the robustness and breadth of coverage that is needed for the successful parsing of unrestricted text or discourse. Parsers developed in the 1970s and 1980s and tested on sets of laboratory sentences were often assumed to handle comprehensively virtually all the possible sentences of the language, until they were tried out on a piece of unrestricted text – say a corpus sample from a national newspaper – and it was discovered how limited their coverage was![2] Hence the importance of a syntactically annotated corpus. With an annotated corpus,

- we can **train** a parser by exposing it to a **training corpus** containing the data of real language in all its diversity and complexity; and
- we can **test** a parser by evaluating its performance against a given corpus as **testbed**.

A typical situation in the recent past has been one in which a corpus annotated by a combination of human and machine processing is used to train and test an automatic parser. Clearly, the success of a parser can be measured by the extent to which it is able to match or replicate the sentence analyses found in the testbed (Black 1993, Brill 1993, and many other studies in the US). One major emphasis in recent years has been in the development of **probabilistic parsers** (Garside *et al.* 1987, Black *et al.* 1992, Briscoe and Carroll 1993, Briscoe 1994) which are pre-eminently robust in being able to parse rare or aberrant kinds of language, as well as more regular, run-of-the-mill types of sentence structures. Such parsers can be trained inductively, that is, through the extraction of frequency counts based on rules, constituent types, and the number of applications or occurrences of these found in a suitable corpus. Given a suitable corpus, probabilistic parsers can also be trained more precisely to parse particular genres or sublanguages, such as the language of computer manuals (Black *et al.* 1993).

With the new emphasis on testing parsers against real text data, rather than 'laboratory sentences', success in parsing has increased to the extent where research teams are claiming success in parsing 70–80 per cent of the sentences in a corpus.[3]

2. **Extracting Lexical Information** A corpus with syntactic annotation can be a rich source of information which can be used to build up and enhance a computer **lexicon**.[4] Along with grammars, lexicons are the basic constructed resources which NLP needs in order to analyse the morphology, syntax or semantics of an input sample of the language. Like large corpora, large lexicons contain a mass of detailed information: and yet, like grammars, they are often woefully incom-

plete (indeed, a complete lexicon is virtually an impossibility). Lexicons can be made more comprehensive with the help of corpora, for example, by adding new rare lexemes (or lemmas); by adding information about their morphological variants; or about the syntactic subcategorization frames of verbs and other parts of speech.[5] But, in addition to all these types of information, a corpus-based lexicon can contain information about the relative frequency of words, lemmas, collocations, case frames, etc., and the distribution of these features in different kinds of text. With the help of a corpus, a lexicon can also be trained or adapted to a particular **sublanguage** or **text type**. However, with a raw or grammatically tagged corpus, most of these types of information can be extracted only by manual methods, using the human mind and hands as intermediaries. With a syntactically annotated corpus, on the other hand, such information can be added semi- or fully automatically, through the matching of corpus annotations with syntactic specifications in the lexicon.[6]

3.2 Treebanks and Skeleton Parsing

Compared with grammatical tagging, syntactic annotation tends to lack a sense of standard practice (see Section 16.4.1). However, what we termed the 'baseline model' above – the PS model – has been widely adopted, particularly for English. It will be appropriate, then, to begin with this relatively simple model, as a means of imparting first-hand familiarity with the field.

The syntactic annotation projects undertaken in the later 1980s and early 1990s built up large corpora by means of fast manual interactive techniques. The need was for a treebank (i.e. parsed corpus) of considerable size – say, 1–3 million words – to act as a training corpus for probabilistic parsers (Garside *et al.* 1987, Black *et al.* 1992, Marcus *et al.* 1993). Since automatic parsing had not yet reached a sufficiently successful stage of development, a type of annotation simple enough to be input speedily by human 'treebankers' was needed, and this led to a **skeleton parsing** scheme based on a shallow PS model of syntactic structure. The Lancaster/IBM treebank (described in Leech and Garside 1991) may serve as an example. The skeleton parsing model was used for parsing about 3 million words of text, of which one smallish section, consisting of the parsed text of the Lancaster/IBM Spoken English Corpus, is available for general research use and scrutiny.[7] In Box 3.1 is a brief sample.[8]

It will be noted that the number of non-terminal symbols (i.e. bracket labels) used is relatively small. Also, the tree is incomplete: in some cases

Box 3.1 A sample of skeleton parsing from the Lancaster/IBM Spoken English Corpus

```
SJ06 298v
[S But_CCB ,_, [[N the_AT thing_NN1 N][V was_VBDZ V]] ,_, [N you_PPY
N] often_RR [V found_VVD [Fn that_CST [Fa although_CS [N you_PPY N][V
had_VHD [N a_AT1 reserved_JJ seat_NN1 N]V]Fa] ,_, that_CST there_EX
just_RR [V would_VM n't_XX be_VB0 [N room_NN1 N][P on_II [N the_AT
train_NN1 N]P]V]Fn]V] ._. S]
```

the brackets are left unlabelled, and this was allowed in order to give annotators a chance to decide that some sequence of words was a constituent, without committing themselves to a label. The simplicity of the scheme was intended not only to speed up the process of treebank compilation, but also to limit the intellectual complexity of the task of human parsing, so that inconsistencies and inaccuracies in treebanking practice were minimized.

Just as grammatical tagging depends on the clear definition of a tagging scheme (see Section 2.1), so syntactic annotation depends on a clear specification of what has been called a **parsing scheme** (Sampson 1995: 4), which has three components similar to those of a tagging scheme:

1. A list of symbols used in the annotation: non-terminals, terminals, and other symbols
2. A basic definition of the symbols: e.g. N = Noun Phrase
3. A description, as detailed as possible, of how the symbols are actually applied to text sentences. For example, how do annotators recognize a Noun Phrase when they see one, and how do they distinguish Noun Phrase tokens from words or word sequences which are not Noun Phrases?

For the Lancaster/IBM Parsing Scheme, (1) the non-terminal labels and (2) their definitions are given in Box 3.2 (overleaf) (the word tags of the C7 tagset are used here as pre-terminals – see Appendix III).

What is more difficult and time-consuming is (3), the description of how the symbols are actually used in the annotation. One possibility would be to let this be specified by a detailed PS grammar. However, our experience was that a comprehensive grammar was (a) difficult for annotators to keep track of, and (b) difficult to update. In practice, the more data is analysed, the more rules are needed – and, contrary to expectation, this need to augment the grammar by additional rules is found to continue, even after a million or more words have been parsed. What is needed is a set of guidelines, which Sampson (1987b: 90–4) likens to the case law of a legal system, based on the continuing refinement of what is learned from

Box 3.2 List of UCREL skeleton parsing symbols

Non-terminal category	**Symbol**	**Example**
Adverbial Clause (Finite)	Fa	[Fa *When it arrived* Fa]
Comparative Clause (Finite)	Fc	[Fc *than we could* Fc]
Nominal Clause (Finite)	Fn	[Fn *that the gas was leaking* Fn]
Relative Clause (Finite)	Fr	[Fr *which nobody wanted* Fr]
Genitive	G	[G *the champion's* G]
Adjective Phrase	J	[J *extremely remote* J]
Noun Phrase	N	[N *the pipes* N]
Metalinguistic Constituent	Nn	*the* [Nn *get directory* Nn]*request*
Temporal/Adverbial Noun Phrase	Nr	[Nr *that day* Nr]
Non-temporal Adverbial Noun Phrase	Nv	[Nv *twenty metres* Nv]
Prepositional Phrase	P	[P *in the next window* P]
Sentence (including direct speech quotation)	S	[S *That's okay* . S]
Interpolated or Appended S	Si	[Si *she said* Si]
-ing Clause (non-finite)	Tg	[Tg *buying it too often* Tg]
To- Infinitive Clause	Ti	[Ti *to send them all away* Ti]
Past Participle Clause	Tn	[Tn *driven by hunger* Tn]
Verb Phrase	V	[V *was coming home* V]
Initial conjunct	&	[N& *High winds* N&] *and*
Non-initial conjunct	+	*and* [N+ *heavy seas* N+]
Discontinuity marker	@	[J *more foggy* J]@ *today* @Fc *than I ever remember* Fc]

individual cases – in fact, the evolving law of precedent, rather than a set of unchanging hard-and-fast rules. This case law is recorded and updated in an in-house **annotators' manual** (referred to as a **tagging manual** in Section 2.1). The guidelines increase and are successively modified during the course of annotation, as more data is analysed and as new problems of how to map parses on to sentences arise. When the project is finished, the guidelines, which have so far been for the use of the annotators themselves, should be edited and consolidated into a document to be available to users of the treebank. Sampson's parsing scheme for the SUSANNE Corpus has been published in a book form (Sampson 1995), from which a brief set of excerpts will give an idea of the meticulous treatment of detail required if the parsing scheme is to be a good guide for users, and for future annotators who might want to adopt the same scheme. In fact the extracts in Box 3.3 form an abbreviated and shortened account of how Sampson proposes to deal with punctuation. The choice

Box 3.3 Excerpt from Sampson (1995: 177-8) showing detailed guidelines, omissions are signalled by '...'

> **§4.63** As we saw in §§2.36 ff., most individual punctuation marks are treated as separate "words" having their own leaf nodes in a SUSANNE parsetree ...
>
> **§4.64** When a punctuation mark (such as a comma or bracket) marks the boundary of a tagma, it is parsed as the sister (rather than first or last daughter) of that tagma. ...
>
> **§4.65** Where a punctuation mark can equally well be regarded as marking the boundary of superordinate or subordinate tagmas, it is treated as bounding the former; i.e. punctuation marks are attached as high in the parsetree as possible. ...
>
> **§4.66** Some punctuation marks (e.g. all brackets, probably a majority of dashes, many commas) occur in balanced pairs, marking either boundary of a tagma. ... Where punctuation marks are paired in this way, the parsing wherever possible makes the members of the pair sisters (that is, ICs [immediate constituents] of the same higher tagma). ...
>
> **§4.67** In some cases, usually with commas, it is unclear whether the punctuation mark occurs as half of a balanced pair or as a "singleton" independently of other punctuation. If the meaning of the sentence, and the norms of orthography, leave this issue genuinely open, then it is settled in terms of the principle of attachment as high as possible. ...
>
> **§4.68** Often, though, decisions about comma placement can be made in terms of more specific considerations ...

of this topic is actually of some instructive interest in itself, showing that compilers of treebanks typically take punctuation to be part of the data they have to incorporate into a parse tree.

Another way to provide details of the parsing scheme is to compile a **reference** or **benchmark** treebank, consisting of parsed sentences taken from a larger treebank, and selected to be illustrative of different features of the analysis. Then the user can search on particular symbols or symbol combinations, in order to induce from the examples the way the corpus has been annotated. This method has been used by the Helsinki group for their ENGCG parsing scheme – see Section 3.3.5. Ideally, the examples and the explicit guidelines should both be available, and there should be thorough cross-referencing between them.

It is worthwhile considering briefly: What is the point of this rather elaborate documentation of the corpus parsing scheme? Admittedly, not

all treebanks are accompanied by this documentation, certainly not in the detail recommended. But there is no harm in a counsel of perfection where current practices frequently fall so far short of the ideal.

If someone uses the treebank for, say, training and testing a parser, they will want to know that the treebank is as accurate and as consistent as possible in the annotations it assigns to a corpus. Accuracy and consistency are interconnected measures (see further, Chapter 17). Annotation is **accurate** to the extent that it conforms to the 'correct annotation'. But what is 'correct'? There is no God's truth in annotation. Instead, the annotator needs to adopt some *de facto* standard of correctness – perhaps initially derived from a grammar, or some previously annotated corpus, or some 'standard guidelines' such as those of EAGLES (see Chapter 16). Unless some such standard is set up, annotation could be assigned on some arbitrary, *ad hoc* or random basis. It is in order to establish some such *de facto* standard that we need a detailed statement of the parsing scheme – which is, in fact, a statement of what is correct for the purposes of the set of syntactic annotation tasks for which that scheme is adopted.

Consistency, on the other hand, measures the extent to which annotations conform to the same standard. The more detailed the parsing scheme, the more it enables different human annotators to be consistent in their analysis of the data. (This consistency may be tested, for example, by seeing how similarly different annotators parse the same set of sentences by hand, or how similarly they correct the same automatically-annotated sample of sentences – see Chapter 17.) Now, the more explicit and detailed the parsing scheme, the more it enables accuracy and consistency to be tested. The difficulty with an unspecific parsing scheme is that it prevents both accuracy and consistency from being demonstrably achieved or properly evaluated.

However, we have assumed above that it is possible to be completely explicit about the 'gold standard' by which sentences are parsed and the parses evaluated. From another point of view, it can be argued that annotation practices should allow for indeterminacy. When treebankers are uncertain about how to parse a sentence, it may be because the sentence itself is genuinely ambiguous, or because there is a legitimate 'grey area' between one category and another. An example of an ambiguous sentence (from the British National Corpus) is:

> The main global-warming gas [...] is carbon dioxide, given off by burning fossil fuels.

The last three words might be analysed as a gerundival *-ing* clause, or as a noun phrase, leading to the following alternative skeleton parses:

[Tg burning_VVG [N fossil_NN1 fuels_NN2 N]Tg]
[N burning_JJ [fossil_NN1 fuels_NN2]N]

although the sense of the sentence appears to make the second analysis more likely.

3.3 Different Varieties of Syntactic Annotation

In the remainder of this chapter, we will illustrate the variety of syntactic annotation practices, by showing examples of the syntactic annotation of English. (At the time of writing, there are few examples available of the annotation of other languages, although this situation is likely to change rapidly.[9])

3.3.1 The Penn Treebank: Phase 1

Today, the largest and best-known treebanking operation is that of Mitchell Marcus and his team at the University of Pennsylvania (see Marcus *et al.* 1993). Because of the location where it originated, the treebank assembled by Marcus is informally termed the **Penn Treebank**. Initially, the method adopted when the project started (around 1990) was closely modelled on that of the Lancaster/IBM Treebank, already discussed in Section 3.2. The method was known as 'skeletal' (rather than 'skeleton') parsing, because, like the Lancaster/IBM project, it did not aim at a complete parse. A PS model of parsing was adopted, but partially parsed trees were accepted into the treebank, because the parser which undertook the task prior to human post-editing (Hindle's Fidditch parser – Hindle 1983, 1989) left prepositional phrase attachments unspecified in many cases. Interestingly, the underspecification of preposition attachment is also found in the Helsinki ENGCG parsing (see Section 3.3.5), and Hindle and Rooth (1993: 114) have reported that between 12 and 15 per cent of prepositional phrase attachments cannot be consistently and correctly resolved even by a human judge, who can make use of contextual and real-world knowledge as well as linguistic knowledge. So there is an argument that syntactic annotations can be overdeterminate.

The following example (Box 3.4) (overleaf) of a very simple sentence from the Penn Treebank (Phase 1) will be easy to interpret, given its similarity to the example of the Lancaster/IBM Treebank in Section 3.2. There is a difference of layout, however, in that the Penn tree is displayed vertically, with indentations recording the depth of branchings in the tree. However, this is a superficial difference, since it would be easy to convert

either one of these formats to the other by a trivial automatic procedure.

Box 3.4 A sentence from the Penn Treebank (Phase 1)

```
( (S (NP (NP Pierre Vinken)
    ,
    (ADJP (NP 61 years)
      old
    ,))
    will
    (VP join
  (NP the board)
  (PP as
    (NP a nonexecutive director))
  (NP Nov. 29)))
  .)
```

A key advantage of the Penn Treebank is that it is generally available throughout the world (though at a cost), through the Linguistic Data Consortium, an American organization that specializes in the acquisition and distribution of corpora of many kinds. It is therefore undoubtedly the most used treebank anywhere in the world at present. The latest information (February 1997) was that the Penn Treebank consisted of 3,300,000 words; apart from the Brown Corpus, which constitutes over 1,000,000 words of that amount, the text types represented are somewhat limited, the *Wall Street Journal* playing a prominent part.

3.3.2 The Penn Treebank: Phase 2

Phase 2 of the Penn Treebank compilation, which is now underway, is much more ambitious than Phase 1 in the amount of information included in parse trees. Once the existence of the Penn Treebank had given researchers in the US an appetite for testing parsers against syntactically annotated data, they sought more detailed information about each sentence: the kinds of information which a parser needs in order to make a full analysis of a sentence, and which a treebank therefore needs if it is to be a fuller and more detailed testbed for evaluating parser performance.

In the Phase 2 treebank, a range of additional information, mostly relating to the deeper or 'logical' level of syntax, is being added:

(a) Functional labels for constituents (e.g. subject, object), as well as categorial labels

(b) Null constituents, or traces (e.g. the 'missing subject' of *to eat* in *It's easy to eat*)
(c) Indices of co-reference (e.g. the co-reference of *Mary$_i$ saw herself$_i$ in the mirror*)
(d) Unusual types of coordination, such as right-node raising and gapping
(e) Discontinuous constituents (or pseudo-attachment)
(f) Semantic roles (e.g. agent, patient, recipient, goal)
(g) Types of adverbial (e.g. temporal, locative, manner)
(h) Syntactic ambiguities, when there is a need to record them.

It remains to be seen whether the new enrichment of the parsing scheme of Phase 2 will be consistent with the production of quantities of data as large as were produced in Phase 1. There is bound to remain a major role for manual analysis in annotation of this degree of complexity.

3.3.3 Nijmegen Treebanks

Long before the Penn Treebank got off the ground, there had been steady growth in treebanking activities at a European centre of corpus activities, the Catholic University of Nijmegen, Holland. Under the leadership of Jan Aarts, the parsing system known as TOSCA (= Tools for Syntactic Corpus Analysis) was set up in the early 1980s. Using a grammatical model known as Affix Grammar,[10] and generating a parser for this, the team assembled two sizeable corpora of English, an initial 'pilot' treebank of 130,000 words known as the **Nijmegen Corpus**, and a much larger treebank, known as the **TOSCA Corpus**, of which 1,000,000 words are undergoing analysis (see van Halteren and Oostdijk 1993). The TOSCA treebank is integrated with the LDB (or Linguistic DataBase), a database facility which provides the means for the treebank to be searched for varied features, to be quantitatively analysed, and to be modified where required (van Halteren and van den Heuvel 1990). One feature of the Nijmegen treebank work is that the automatic parsing interacts on-line with the human annotator, who is able to resolve ambiguities by hand, through a sophisticated user interface, or to change the parse where necessary. This is another variant of the division of labour between human and machine analysis which is observed in various forms in different corpus annotation projects. (See Chapter 11 for a further variation on this theme, in respect to treebank development.)

From the example given in Box 3.5 (overleaf) of a sentence from the TOSCA treebank we note a family resemblance to the Lancaster and Penn treebank examples already seen. The Affix Grammar generates a phrase structure model where attributes can be attached to each node. The lay-

Box 3.5 Sentence from the TOSCA Treebank

```
-:TXTU( )
  UTT:S(act,indic,inter,motr,pres,unm)
    INTOP:AUX(do,indic,pres){Does}          Does
    SU:NP()
      NPHD:PN(pers,sing){he}                he
    V:VP(act,do,indic,motr)
      MVB:LV(indic,infin,motr){realize}     realize
    OD:CL(act,indic,intens,pres,unm,zsub)
      SU:NP()
        NPHD:PN(pers,sing){he}              he
      V:VP(act,indic,intens,pres)
        MVB:LV(indic,intens,pres){is}       is
      CS:AJP(prd)
        AJHD:ADJ(prd){wrong}                wrong
  PUNC:PM(qm){?}                            ?
```

out in Box 3.5 with its indentations recording the step-by-step branching of a PS tree, resembles in this respect the Penn Treebank sentence in Box 3.4. Four kinds of information can be attached to a node, as illustrated in this sentence. First, a syntactic function (SU 'Subject'; OD 'Direct Object'); secondly, a category label such as MVB (= main verb); thirdly, a set of attribute labels (such as motr 'monotransitive' as an attribute of a verb); fourthly the lexical content of each leaf node: that is, the word form itself, e.g. {realize}.

The Nijmegen Affix Grammar, as a descriptive grammar of English, is ultimately founded on the descriptive grammars of Quirk *et al.* (1972, 1985), and one of its claims to attention is that it is the first formalization of the syntax of the language based on such detailed reference grammars, which are arguably a better starting-point for wide-coverage corpus parsing than more 'elegant' theoretical grammars.

3.3.4 The SUSANNE Corpus

Geoffrey Sampson's SUSANNE Corpus (see Sampson 1995) is a treebank which, although containing only 128,000 words, provides a great deal of parsing information for each sentence. Like a richly illuminated medieval manuscript, it contains much detail within a small compass. Also like a medieval manuscript, it is 'hand-crafted' – being the result of manual analysis.[11] Another manifestation of the great care lavished on this treebank is that the parsing scheme is specified in great detail (see Section 3.2). This includes grammatical tagging decisions, too. A further point in favour of the SUSANNE Corpus is that it is distributed for general world-

Box 3.6 An example from the SUSANNE Corpus. (From Sampson 1995: 32.)

N03:0460f	–	YB	<minbrk>	–	[Oh.Oh]
N03:0460g	–	PPHS1m	He	he	[O[S[Nas:s.Nas:s]
N03:0460h	–	VVDt	handed	hand	[Vd.Vd]
N03:0460i	–	AT	the	the	[Ns:o.
N03:0460j	–	NN1c	bayonet	bayonet	.Ns:o]
N03:0460k	–	II	to	to	[P:u.
N03:0460m	–	NP1m	Dean	Dean	[Nns.Nns]P:u]
N03:0460n	–	CC	and	and	[S+
N03:0460p	–	VVDv	kept	keep	[Vd:Vd]
N03:0460q	–	AT	the	the	[Ns:o.
N03:0470a	–	NN1c	pistol	pistol	.Ns:o]S+]S]
N03:0470b	–	YF	+.	–	.

Key: AT = *the*; c = common; CC = co-ordinating conjunction; II = preposition; m = masculine; minbrk = minor break; N = noun phrase; NN1 = noun; NP1 = proper noun; O = paragraph; :o = logical direct object; P = prepositional phrase; PPHS1 = singular personal pronoun; s = singular; S = main clause; :s = logical subject; t = transitive verb; :u = prepositional object; v = transitive/intransitive verb; Vd = verb group beginning with past tense; VVD = past tense of transitive verb; YB = text division of paragraph or higher rank; YF = full stop (Sampson 1995: 105–20, 168–70, 362).

wide use: in its creator's words, 'it is now available to the research community freely and without formalities' (Sampson 1994).[12] Against these positive points, it must be conceded that the SUSANNE Corpus consists of texts more dated than most people would ideally like to consult: it is a subset of four genres of the Brown Corpus, first collected in 1961.[13]

Sampson began his work on corpora while at Lancaster in the mid-80s, so the SUSANNE Corpus is ultimately from the same stable as the Lancaster and Lancaster/IBM Treebanks. Not surprisingly, a similar type of analysis is employed, modelled on a PS grammar with attributes on nodes. But SUSANNE also contains much more detailed information, in such areas as syntactic functions and deep structure (including null constituents and co-reference indices). The brief example in Box 3.6 shows a vertical layout in the form of a set of fields or columns, in which each row represents a (word-)token.

In this format, different kinds of information are represented in different columns. The first (leftmost) column shows location references, the second column shows wordtags, the third word forms, the fourth lemmas (the words in their base form), and the fifth the PS parse, in the familiar labelled-bracketing notation.

3.3.5 The Helsinki Constraint Grammar

The last kind of treebank we will discuss represents a thorough-going break with the common traditions observed in most other treebanks. The only connection worth mentioning is that like the Nijmegen corpus parsing, the Helsinki corpus parsing acknowledges a debt to the style of descriptive grammar represented by Quirk *et al.* (1972, 1985). Other comparisons must emphasise differences.

The first difference to be mentioned is that the term 'treebank' is hardly applicable to the output of a Helsinki constraint grammar parser (described in Voutilainen *et al.* 1992, Karlsson *et al.* 1995). This is for two reasons. First, the emphasis of the Helsinki team, led by Fred Karlsson and Atro Voutilainen, is on the techniques of processing corpus data, rather than on the product of such processing, in the form of an annotated corpus. To our knowledge, none of the texts or corpora processed at Helsinki have been publicly distributed, although the Helsinki English Constraint Grammar parser ENGCG has been applied to the largest English corpus of all – the Bank of English Corpus (of more than 300 million words). Secondly, the method of analysis applied departs from the PS grammar model, and adopts a dependency grammar model instead. A dependency grammar models the structure of a sentence in terms of a tree of which the nodes are words, rather than constituents of different sizes. Thus a typical dependency representation of a noun phrase such as *a quarterly bulletin on employment* will show the configuration in Figure 3.1, whereas the corresponding phrase structure tree will be as in Figure 3.2.

The ENGCG parser does not provide a complete dependency parse, however: 'dependent' nodes are left unattached to a higher 'governor' node, although the direction of the attachment is indicated and the 'governor' can usually be inferred. In addition to providing dependency information, the parser gives a detailed breakdown of the attributes of individual words, including, for example, subcategorization (valency) information for verbs. Functional labels such as 'subject' and 'object' are also added, although (ENGCG being a dependency model) such labels are

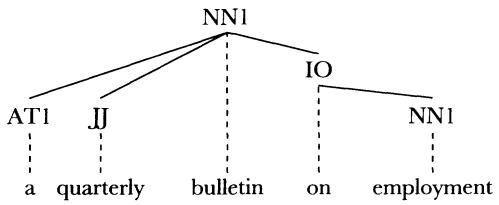


Figure 3.1 A dependency tree

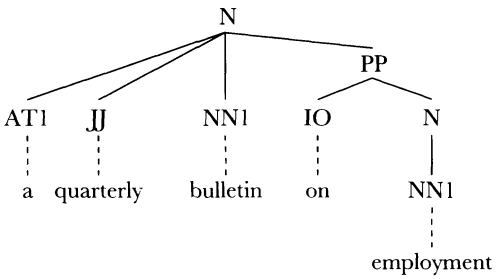


Figure 3.2 A phrase structure tree

attached to words rather than phrases or clauses. A brief sample of Helsinki parser output is given in Box 3.7 (overleaf). As can be seen, for each word the annotation is placed on a separate line below, beginning with the lemma, and ending with syntactic labels (those marked by @). The example of *less than* (lines 14–17) shows three different analyses, where ENGCG has failed to disambiguate. Most tokens are given a single analysis. Dependency analysis is indicated by < preceding the syntactic label or > following it; e.g.: <P (line 13) marks a complement dependent on a preceding preposition, AN> (line 2) marks an adjective dependent on a following nominal. Other syntactic labels mark verb chain members (e.g. +FMAINV 'finite main verb') and nominal heads (e.g. SUBJ 'subject').

ENGCG is still undergoing development, but is already a highly impressive piece of software. In the amount of detail it provides, it is almost in the same league as the Penn Treebank (Phase 2) and the SUSANNE Corpus. However, it must be admitted that the term 'parser', as applied to this software, is something of an overstatement: the performance is somewhere between a tagger and a parser. On the other hand, the processing, in contrast with other software mentioned here, is conceived of as entirely automatic, with an optional phase of correcting errors or (more typically) eliminating ambiguities after the automatic parser has done its work.

The Helsinki Constraint Grammar parsing is also being applied to texts in other languages, such as Finnish, Swedish, German, Danish and Portuguese. The choice of a dependency model is thought to be particularly appropriate for application to inflectional or agglutinative languages such as those mentioned. It is arguable that English, as a language with impoverished morphology, is not a suitable model on which to base the parsing of other languages, particularly other European languages, which have a rich morphology. Therefore, the treebank model that has been so vigorously developed in the analysis of English corpora may after all turn out to be less suitable than a dependency-based model for such languages.

Box 3.7 Output from the Helsinki ENGCG parser, representing an analysis of the sentence *Royal Dutch Shell, worth just $500m less than Exxon, is third.* (From Karlsson *et al.* 1995: 403.)

```
("<*royal>                                                          1
  ("royal" A ABS (@AN>)))                                           2
("<*dutch>"                                                         3
  ("dutch" <Nominal> A ABS (@AN> @<NOM)))                           4
("<*shell>"                                                         5
  ("shell" N NOM SG (@SUBJ)))                                       6
("<$,>")                                                            7
("<worth>"                                                          8
  ("worth" PREP (@ADVL)))                                           9
("<just>"                                                          10
  ("just" ADV (@AD-A>)))                                           11
("<$500m>"                                                         12
  ("$500m" NUM CARD (@<P)))                                        13
(<"less=than>"                                                     14
  ("less=than" <CompPP> PREP (@ADVL))                              15
  ("less=than" <**CLB> CS (@CS))                                   16
  ("less=than" ADV (@ADVL)))                                       17
("<*exxon>                                                         18
  ("exxon" <Proper> N NOM SG (@<P)))                               19
("<$,>")                                                           20
("<is>"                                                            21
  ("be" <SV><SVC/N><SVC/A> V PRES SG3 VFIN (@+FMAINV)))            22
("<third>"                                                         23
  ("third" NUM ORD (@PCOMPL-S)))                                   24
("<$.>")                                                           25
```

3.3.6 Comparative summary

The above is not a complete overview of the syntactic annotation projects and treebanks that exist. Just limiting attention to English, we have omitted one significant example, the POW (Polytechnic of Wales) Corpus, a treebank of children's language, annotated according to a scheme based on Halliday's systemic functional grammar (Fawcett and Perkins 1980, Souter 1989 – see Halliday 1985). But our purpose has been to give a brief yet broad survey of syntactically annotated corpora of English, with appropriate reference to the procedures used in annotation. In conclusion, it will be useful to classify the various layers of information which we have noted and which may be provided by a treebank, as a basis for a tentative treebank typology (see Table 3.1; see further Leech *et al.* 1995).

Table 3.1 Layers of Information in a Treebank[a]

		LI	P1	P2	TO	SU	He
(1)	Bracketing of Constituents	++	++	++	++	++	–
(2)	Labelling of Constituents	++	++	++	++	++	–
(3)	Dependency Relations	–	–	–	–	–	+
(4)	Functional Labels	–	–	++	++	++	++
(5)	Subcategories/Attributes	+	+	+	++	++	++
(6)	Deep/Logical Structure	–	+	++	–	+	–
(7)	Spoken Language Features	–	–	–	–	–	–

[a] based on Figure 9 in Leech, Barnett and Kahrel (1995)

Key: *Corpora*
LI = Lancaster/IBM Treebanks
P1 = Penn Treebank, Phase 1
P2 = Penn Treebank, Phase 2
TO = TOSCA Corpus, Nijmegen
SU = SUSANNE Corpus
He = Helsinki ENGCG

Amount of information given for a given layer
\+ Some information
++ A good deal of information
– No information
[The above ratings are impressionistic]

Of the different layers, (1) and (2) in Table 3.1 are those found in the basic 'skeleton treebank'. Dependency relations (3) are found only in the Helsinki parsing. Functional information (4) is found in all except the skeleton/skeletal treebanks of Lancaster/IBM and Penn Phase 1. Subcategorization/attributes (5) are interpreted as applying to non-terminal constituents of PS trees, rather than to the terminals or grammatical wordtags; to some extent, they are found in all the mentioned treebanks. Layer (6) represents the various kinds of semantically-oriented information highlighted in the discussion of Penn Phase 2 (Section 3.3.2).

The last row of Table 3.1 (layer 7) is empty: although the Lancaster/IBM SEC consists of spoken language, it is the kind of spoken language which is typically edited and corresponds most closely to written style. It is regrettable that very little experience has so far been gained of the problems of treebanking spontaneous speech. However, things are now changing rapidly in this regard. A project has been completed at Lancaster, in which 24,000 words of the spoken data of the BNC have been skeleton parsed (Eyes 1996), to match a similar amount of written data from the same corpus. For this, it has been necessary to devise special symbols to represent the occurrence in parse trees of the pervasive non-fluency features of conversation, such as incomplete utterances, unplanned repetitions and false starts, as well as utterances which contain 'unclear' passages in the transcription. A similar project has taken place at Nijmegen, where the TOSCA parser has been undergoing adaptation to conversational language. At the University of Sussex, Geoffrey Sampson

has recently begun work on a new extension of his SUSANNE project, to include spoken as well as written language. Most impressive of all, the Penn Treebank (according to a recent report) contains a million words of spoken data, the dysfluencies of which have been annotated.

There is no strict order of precedence among the layers of Table 3.1: for example, in existing treebanks, layers 1 and 2 do not combine with layer 3. None the less, there is a loose sense in which (1)–(7) show how annotation can climb up a ladder from basic information to more sophisticated, abstract or complex information. It is difficult to be sure which of these layers will prove most important in future research, but at present the decision as to which layers to concentrate on must rest on such practical factors as the money and human time available, and the applications to which the treebank is expected to contribute.

This chapter has been introductory, and has left many issues and details of syntactic annotation unexplored. Further surveys of the field are provided by Sampson (1992), Souter (1993), Souter and Atwell (1994) and Leech *et al.* (1995), the latter two of these focusing on the issues of standards and standardization. In this book, Chapter 11 takes further the investigation of treebanks and syntactic annotation.

Notes

1. Conceptually, the most straightforward type of PS annotation is the labelled bracketing associated with a context-free phrase structure grammar (CFPSG). However, it should be noted that more advanced and sophisticated models, based on PS models, have recently been applied to corpus parsing and annotation. They include probabilistic CFPSGs, unification grammars, and other grammars in which non-terminal nodes are associated with feature-attribute complexes.
2. The problems of earlier generations of parsers, and the advances made by statistical and corpus-based methods, are well presented in Marcus (1995), part of whose abstract reads as follows:
   'The field of natural language processing (NLP) has seen a dramatic shift in both research direction and methodology in the past several years. In the past, most work in computational linguistics tended to focus on purely symbolic methods. Recently, more and more work is shifting toward hybrid methods that combine new empirical corpus-based methods, including the use of probabilistic and information-theoretic techniques, with traditional symbolic methods. This work is made possible by the recent availability of linguistic databases that add rich linguistic annotation to corpora of natural language text. Already, these methods have led to a dramatic improvement in the performance of a variety of NLP systems with similar improvement likely in the coming years.' (Marcus 1995: 10052)

3. In fact, measuring correctness of parses and parsers is more complicated than measuring that of grammatical tagging. A high percentage or a low percentage can be obtained by the same parsing experiment, according to how the success rate is calculated. For some detail on this, see Black *et al.* (1993: 2–5)
4. On computer lexicon development, see Boguraev and Briscoe (1989) and Pustejovsky (1995).
5. On subcategorization and valency in the computer lexicon, see Sanfilippo *et al.* (1996).
6. The use of annotated corpora for lexicon development and enhancement, although still in its infancy, is likely to become important in the future. The coming together of corpus annotation and syntactic information in the lexicon is evident in Sanfilippo *et al.* (1996), and also in the present European Union's PAROLE2 project, which undertakes the development of corpora for different European languages alongside the development of lexicons.
7. The Lancaster/IBM SEC treebank is available from the Norwegian Computing Centre for the Humanities (see Appendix I). There is also a CD-ROM database version of this corpus, known as MARSEC: see Knowles (1993, 1995).
8. As with grammatical tagging, so with syntactic annotation, there is the possibility of translating an everyday format such as that in Box 3.1 into an interchange format, using TEI guidelines. To give an idea of how a TEI-conformant skeleton parse would look, the following is a simplified example (based on Sperberg-McQueen and Burnard 1994, Section 26.4.2). The <s> tag may be read 'sentence', and the <c> tag 'constituent'. On the right is the same parse as represented in the UCREL-style format of Box 3.1, except that the vertical format of the TEI-conformant example is retained, including the use of indentation to show hierarchical constituent structure.

```
<s type=sent>                          [S
<c struct=N>                           [N
  <c struct=G> The victim's </c>         [G The victim's G]
  friends                                friends
</c>                                   N]
<c struct=V>                           [V
  told                                   told
  <c struct=N> police </c>               [N police N]
  <c struct=Fn>                          [Fn
    that                                   that
    <c struct=N> Krueger </c>              [N Krueger N]
    <c struct=V> drove                     [V drove
      <c struct=P> into                      [P into
        <c struct=N> the quarry </c>           {N the quarry
                                               N]
      </c>                                   P]
    </c>                                   V]
  </c>                                   Fn]
</c>                                   V]
</s>                                   S]
```

9. Examples include a treebank of French text undertaken at IBM Paris (by J.-M.

Langé and others); and various syntactic annotation projects in progress using the Helsinki Constraint Grammar (see Section 3.3.5).

10. To be more precise, two versions of Affix Grammar have been used in the compilation of the TOSCA corpus: (a) extended Affix Grammar, and (b) Affix Grammar over finite lattices (see Nederhof and Koster 1993).
11. It is intriguing that Sampson himself calls the SUSANNE parsing scheme 'a Domesday Book of English grammar', and, in another historical simile, likens it to a 'Linnaean taxonomy'.
12. The SUSANNE Corpus is available from the Oxford Text Archive by anonymous ftp (see Appendix I).
13. The reason why Sampson used texts from the Brown Corpus is that he was able to obtain permission to use as the basis for his own treebank the Gothenberg Corpus of Alva Ellegård (see Ellegård 1978), which used data from four text categories of the Brown Corpus. The Gothenburg Corpus was (to our knowledge) the first syntactically-annotated corpus to be created.

4

Semantic Annotation

ANDREW WILSON and JENNY THOMAS

4.1 Introduction

There is often more than one way of referring to an object or a concept. This observation has important implications in a number of areas in which text analysis is undertaken. In discourse analysis, for instance, the choice of how a concept is expressed can reveal information about the ideologies contained in a text or the relationships between participants in conversation. For example, in doctor-patient interaction, one doctor might display a rather formal 'scientific' attitude through the use of more technical terms such as *abdomen*, whereas another doctor might try to relate to the patient on his or her own level by choosing more general, everyday words for the same things, for example *tummy*.[1] Another area in which this notion of choice has implications is information retrieval. For example, a fashion historian might wish to query a large computer text database such as an archive of *The Times* newspaper for articles about changing fashions. But some articles relating to fashions in, say, trousers might not actually use the keyword *trousers*: they might, for instance, refer to *slacks, shorts, leggings, jodphurs, breeches* or even *Oxford bags*! There is also a second issue here. Not only can a single concept be referred to by means of a number of different words: a single word may also refer to a number of different concepts. Suppose that our fashion historian wants to look for articles about boot fashions in the newspaper archive. She might make a query looking for the keyword *boot* in all the articles in the archive. But not all the articles extracted by this query will be about footwear: some will be about computers (*boot* is also a technical term for starting a computer), others will be about cars, and still others will depict violent actions (e.g. *She gave it a boot*). So what we have is a situation where we simultaneously need to make information more general and more specific: we need to be able to identify related words, so that we are not restricted to a few keywords that we can think of, and we need to be able to identify the senses of particular words

in particular contexts, so that we do not extract information that is of no use to us.

One way of solving these two problems is by means of **semantic annotation**. If we attach a label to every word in a text which indicates the semantic field in which it falls, then we can extract all the related words (by querying on the semantic field) and only those instances of ambiguous words with the specific senses in which we are interested (by querying on the combination of word and semantic field).

4.2 Semantic Fields

A **semantic field** (sometimes also called a conceptual field, a semantic domain, a lexical field, or a lexical domain) is a theoretical construct which groups together words that are related by virtue of their being connected – at some level of generality – with the same mental concept. This is not the same as saying that they refer to precisely the same thing or to an opposite of that thing. Such relations – synonymy and antonymy – constitute the more 'delicate' construct of a **lexical set** (cf. McArthur 1986). In contrast to this, a semantic field will also include hypernyms and hyponyms of a word, and additionally words which are associated in other ways with the concept concerned. To give an example, if we assume that we have in our minds a model of some area of human activity – say equestrianism – we can immediately think of a number of words which are connected with that prototypical model. With equestrianism, we would think of words such as *rider, horse, eventing, spurs, saddle, dressage, jump-off* and so on. Most of the words that we would think of would not be synonyms or antonyms of *equestrianism*. In fact, most of them (e.g. *spurs*) would have no sense relationship at all to that word in terms of what they actually mean or denote: rather, they would be related by the fact that they are connected in some way with the same sphere of activity. We say that words which are related in these ways belong to the same semantic field.

4.3 Criteria for Semantic Field Annotation

An annotation scheme for semantic field analysis represents a set of abstract concepts (the semantic fields) under which the words and phrases in the text are classified.

In practice, semantic annotation schemes are something of a compromise between attempts to mirror how words are related in the human mind and the need for usable annotated corpora and reference works by

linguists and other scholars. We have at present very little detail about the content and form of the mental lexicon. For instance, are concepts close clusterings of nodes in a large network of interrelationships, or are they stored in a different way? And exactly how is perceptual reality classified in the mind, i.e. what concepts actually exist? We have some answers – e.g. animacy seems to be a salient conceptual factor – but our knowledge is incomplete and so, until the state of our knowledge advances further, we must progress simply by extrapolating our models on the basis of what seems plausible according to the evidence that we already possess. By classifying words according to a category system representing a set of plausible relationships (i.e. semantic fields) we can approximate to a representation of the kinds of relationships which we know to exist in the mind whilst simultaneously presenting these groups of related words in a way which is maximally accessible to end users of an annotated corpus. Classification of words according to semantic field systems seems the best compromise between what we know about the mind, what is useful for further psycholinguistically-motivated textual research based upon this knowledge, and what other content-oriented scholars and commercial users will find useful and accessible.

As Schmidt (1988) has observed, there is no such thing as an 'ideal' semantic annotation system. However, the following features enter into the equation when choosing or devising a system for adoption:

1. *It should make sense in linguistic or psycholinguistic terms.* This is the most important – and perhaps the most obvious – criterion, but it is worth reiterating since not all existing systems conform to it. If nothing else, a *prima facie* acceptable grouping of words makes an annotated corpus more useful for other users than does a set of more abstract groupings which, although they may be well motivated in theoretical terms, are unhelpful for other kinds of research. We can be fairly sure that certain basic categories exist in the mind, although we cannot be sure exactly how they are structured and at present we do not have an exhaustive set of categories based upon neuropsychological evidence. Future discoveries may lead us to revise our semantic category systems somewhat, but in general there is good agreement between many basic categories in the various systems and the categories we already know about from neuropsychology, for example colours, body parts, topography, and so on. This may give us some degree of confidence that most category systems are being constructed along the right lines, but we should beware of those which clash strongly with our existing evidence and our intuitions. For example, Roget's system (original system 1852; frequently revised) classifies words in the marriage/sexual domain into

groups which also contain general words connected with the concept of 'join': this is clearly an overabstraction and might be positively unhelpful for applications in areas such as information retrieval. Similarly, Dornseiff's classification (original system 1933; later revisions) of *river* may also be questioned: he has suitable geographical categories for seas, lakes, etc., but classifies rivers and similar phenomena amongst more general categories relating to movement.

2. *It should be able to account exhaustively for the vocabulary in the corpus, not just for a part of it* (e.g. speech act verbs or words connected with evaluation). Systems of the latter type are sometimes found within the methodology of content analysis (see Section 4.8) but are unsuitable for more general semantic corpus annotation. If a word or concept cannot readily be classified in the existing annotation system, then the system clearly needs to be amended.
3. *It should be sufficiently flexible to allow for those emendations which are necessary for treating a different period, language, register or textbase.* For example, the treatment of medical texts may require considerably more detailed subclassification of the medical domains than other texts, as users may want to extract information from them at these more specific levels of word and concept relatedness.
4. Related to point (3), *it should operate at an appropriate level of granularity* (or delicacy of detail). This criterion has both a theoretical and a practical motivation. In theoretical terms, it is known from psycholinguistic experiments that words are related conceptually at varying levels of generality. For example, there appears to be a mental concept of 'animal', a concept of 'mammal' and a concept of 'monkey'. Annotation systems should thus, if possible, try to capture these various levels in the conceptual 'hierarchy', which may be important psychologically and thus also in studies of text production and comprehension. In practical terms, to be of most use the annotation system should have a workable number of members in each semantic field which it contains. On the one hand, if the categories are too general, then, when an analyst tries to use a semantically annotated corpus, he/she will find that it only contains a handful of concepts and that within these concepts there exist obvious conceptual groupings that have not been indentified. In other words, if these have not been identified by the annotator, the semantically annotated corpus is reduced practically to the status of a raw corpus, and it is almost useless as a tool for extracting concepts and their wording. On the other hand, if the categories are too specific, then the resulting annotated corpus will again be only a small step up from a raw corpus and more general conceptual relations between words will be veiled. But precisely what level of granularity is correct

for an annotation system is an open question and depends at least partly on the aims of the end user. For this reason, the next criterion is posited.

5. *It should, where appropriate, possess a hierarchical structure.* As we already observed, we can identify a number of levels of generality in concept relatedness. If a semantic category system has a hierarchical structure based on increasingly general levels of relatedness between words and concepts, we can identify all these different levels without imposing a hard and fast decision on which level the end user must employ: the user can look easily at all the different levels, simply by moving up or down to the next level of granularity in the hierarchy. (For a cognitive viewpoint on conceptual hierarchies, see Ungerer and Schmid 1996: 60–113.)
6. *It should conform to a standard, if one exists.* Currently, there is no standard for semantic field systems, but the existence of one would lead to an easier accumulation and comparison of research results for different languages, periods, genres, etc., and may thus perhaps be a logical next step in the development of the field. Such standards need not be in the form of a hard-and-fast system of categories to be applied. This, even if the result of consensual work, may be rejected by many researchers and would, in any case, be unhelpful, as different languages, periods and textbases logically require slightly different distinctions to be made. However, they could, as the EAGLES guidelines have done for the morphosyntactic tagging of modern European languages (cf. Leech and Wilson 1994), lay a broad framework of principles and *major* categories, which will facilitate comparability but which also can be modified as necessary for individual needs.[2]

4.4 Existing Semantic Annotation Schemes

A wide range of systems for semantic field analysis already exist and a number of these are listed in Table 4.1 (overleaf), along with some of the published works and projects in which they have been used.

As already suggested, the majority of these semantic category systems tend to agree, to a greater or lesser extent, on the basic major categories that they contain. With a few exceptions, the main differences between them consist in the structuring of the categories (i.e. the hierarchy) and in their granularity. For example, as Table 1 in Schmidt (1988: 47) demonstrates, the Roget and Hallig-Wartburg-Schmidt systems have similar sets of semantic fields covering the domains of space and geography. However, whereas the Hallig-Wartburg-Schmidt system has only a

Table 4.1 A range of current systems, with a selection of their users

System	Brief description of project and bibliographical reference
Hallig-Wartburg (1952)	*Conceptual Dictionary of Mycenaean Greek* Kazanskiene and Kazanskij (1986); [revised]: *Conceptual Dictionaries of MHG Epic*: Schmidt (e.g. 1993) *Conceptual Dictionary of Gascon* Baldinger (1975); [revised]: *Conceptual Glossary of Latin Vulgate of John's Gospel:* Wilson (1996)
Dornseiff (1933–)	*Conceptual Glossary of Virgil's Eclogues*: Najock (forthcoming)
Louw-Nida (1989)	*Semantic Domain Lexicon of Greek New Testament*: Louw and Nida (1989); *Lexicon of Metal Terminology in Hebrew Scriptures*: Heidebrecht (1993)
Roget (1852–)	*Thesaurus of English Bible*: Day (1992) *Automatic Content Analysis in the Humanities*: Sedelow and Sedelow (1969); *Old English Thesaurus (early stages)*: Roberts (1978)
LDOCE (1978)	*Enriching Corpora with Semantic Information*: Janssen (1990).
Tailor-made	*Historical Thesaurus of English*: Roberts (1994)
Wherle-Eggers (1961)	*None known*

category entitled 'Heavenly Bodies', Roget has more detailed subdvisions such as 'Star', 'Nebula', 'Zodiac', 'Planet', 'Meteor', 'Sun' and 'Moon'. Which of these various hierarchical arrangements and levels of granularity are most appropriate is not a question that can be answered globally: the decision depends on the proposed application.

4.5 Problems in Semantic Annotation: fuzzy sets

Not all words fall conveniently into a single semantic field. We are referring here not to cases of semantic ambiguity, where a word may have more than one sense: rather, we are referring to words which may simultaneously signal more than one concept for the same occurrence in a text. For instance, the word *sportswear* belongs equally in the semantic field of 'Sport' and the semantic field of 'Clothing'. This is related to the fact that semantic relationships between words in the mind appear to behave as so-called **fuzzy sets** (cf. Zadeh 1982) rather than as discrete Aristotelian

categories. Clearly, if a semantically annotated corpus is to be made maximally useful to the end user, we need to capture all the fields into which such 'fuzzy' words fall. One possible way of doing this would be to represent semantic relations between words in the form of a network, as Miller *et al.* (1990/93) have done with their WordNet software. However, although it is possible to capture relationships such as synonymy and antonymy in this way, it is harder to capture some of the more general associative relationships between words which are entailed in semantic field theory. An alternative way of representing multiple membership of semantic fields is by a type of cross-referencing. Some lexicographical works based on semantic fields, for example, Schmidt (e.g. 1993) and Wilson (1996), classify the relevant textwords under more than one semantic field and cross-reference the different categories. However, this kind of cross-referencing is more difficult when tagging a corpus (as opposed to constructing a reference dictionary), since, in corpus annotation, a word is normally assigned only one tag. An alternative but related method, therefore, is to use tags similar in form to the so-called **portmanteau tags** used in morphosyntactic annotation (see Chapters 7 and 9). Here, words with multiple memberships of semantic fields would be assigned tags for each field to which they belong, but these would be assigned in the form of a single tag; for an example, see point 1 in Section 4.6. Where annotated corpora are stored in the form of a relational database, however, it may be more appropriate simply to create multiple links for a given textword to each individual semantic field, rather than to use portmanteau tags.

4.6 An Example of Semantic Annotation

As an example of a semantically annotated text corpus, we present here a portion of text annotated using the revision of Hallig and von Wartburg's (1952) semantic category system, developed by Schmidt (e.g. 1993) and also employed, in a slightly amended form, by Wilson (1996). Although Schmidt and Wilson were working with a full-text relational database and a lexical database respectively, the annotation scheme is also readily adaptable to the physical tagging of text corpora. This system was selected as an example because it conforms very well to the desiderata outlined above: it possesses a hierarchical structure, which also allows some degree of flexibility in extending and reducing the structuring of the categories; it is able to account exhaustively for the vocabulary; it makes *prima facie* sense; and it has a very well-differentiated granular structure.

The tagged corpus extract appears in Box 4.1. The key to the example should be self-explanatory, but two further points to note are as follows:

1. As observed in Section 4.5, one problem with which semantic annotation has to contend is that of multiple membership of semantic fields. In this example, we have treated multiple membership through the use of **'dash tags'**, which perform a similar function to the **portmanteau tags** which are sometimes applied in morphosyntactic annotation. In the 'dash tag', the two semantic tags that may be assigned to the word in context are *both* assigned, but in the physical form of a single tag. The two components of that tag are separated by a dash character, hence 'dash tags'. For example, the word *deepened* is given the 'dash tag' 312411-319 (COLOUR–CHANGE/REMAIN).
2. As well as treating single words, it is also often necessary or desirable to treat multi-word units. These include phrasal verbs (such as *stubbed out* in Box 4.1), multi-word noun phrases (such as *riding boots*), proper names (such as *United States of America*), true idioms (such as *living the life of Riley*), and so on. Instead of attempting to analyse these as sequences of individual words, each with their own semantic content, a multi-word unit may be analysed instead as if it were a single word, using a form of **ditto tagging** (cf. Blackwell 1987). Ditto tagging involves the assignment of the same tag to every word in a multi-word unit, together with a marker showing the length of the unit and the position of each word within it. Thus, for example, *stubbed out* in Box 4.1 is tagged as:

   stubbed 21072-31246[m1.2.1
   out 21072-31246[m1.2.2

   where [m1 indicates the first multi-word unit in the corpus, the following 2 indicates that this multi-word unit contains two words, and the third digits indicate that *stubbed* is the first word in the two-word unit and *out* is the second word in the two-word unit.

The hierarchical organization of the category system may also readily be observed, for example, in the tag for Colour (312411), where:

3 = Man And The Environment (highest level in the hierarchy)
31 = A Priori
312 = Characteristics/Conditions
3124 = Characteristics Perceptible By The Senses
31241 = Sight
312411 = Colour

Box 4.1 Sample of semantically tagged corpus

0000001	001	-----	-----	
0000001	010	NP1	Joanna	231112
0000001	020	VVD	stubbed	21072-31246[m1.2.1
0000001	030	RP	out	21072-31246[m1.2.2
0000001	040	APPGE	her	0
0000001	050	NN1	cigarette	2111014
0000001	060	IW	with	0
0000001	070	JJ	unnecessary	317
0000001	080	NN1	fierceness	227052
0000001	081	.	.	
0000002	001	-----	-----	
0000002	010	APPGE	Her	0
0000002	020	JJ	lovely	22706
0000002	030	NN2	eyes	21061
0000002	040	VBDR	were	311
0000002	050	JJ	defiant	228262
0000002	060	II	above	0
0000002	070	NN2	cheeks	2103
0000002	080	DDQGE	whose	0
0000002	090	NN1	colour	312411
0000002	100	VHD	had	0
0000002	110	VVN	deepened	312411-319
0000003	010	II	at	0
0000003	020	NP1	Noreen	231112
0000003	021	GE	's	0
0000003	030	NN1	remark	231212
0000003	031	.	.	

Key The corpus text is read vertically. The first two columns of numerals on the left-hand side are simply reference numbers identifying unique word positions in the text. The third column contains part-of-speech (morpho-syntactic) tags. The next column contains the text words themselves, and the final column contains the semantic field tags. The tags used in this example are:

0	Low Content Words	231112	Personal Names
2103	Body And Body Parts	231212	Linguistic Expression
21061	Organs And Their Functions: Sight	311	Existence/Being
21072	Object-Oriented Physical Activity	312411	Colour
2111014	Luxury Items	31246	Temperature
227052	Anger	317	Causality/Chance
22706	Aesthetic Sentiments	319	Change/Remain
228262	Obedience/Disobedience		

4.7 Assigning Semantic Field Tags

Semantic field annotation is a more difficult task to automate than part-of-speech (morphosyntactic) tagging and also, arguably, than syntactic parsing. This is at least partly because it is inherently knowledge-based (and thus requires a lexicon containing at least a single form of every word in the corpus[3]) and because it is not yet clear that statistical models of transitions between categories, such as are used in probabilistic part-of-speech taggers to achieve accuracy rates in excess of 95 per cent, can be extended to treat semantic field categories.[4] Nevertheless, it is not necessarily the case that semantic field tags have to be assigned manually to texts. In fact, there are three ways in which a text can be annotated with semantic fields:

1. Manual annotation
2. Computer-assisted annotation
3. Fully automatic annotation

Manual annotation requires no further definition. Computer-assisted annotation refers to a semi-automated form of manual annotation, that is, one which is supported by a computer-readable lexicon containing possible semantic fields for given words and perhaps also by a limited amount of automatic disambiguation for certain high-frequency ambiguities. The computer will assign candidate semantic fields to all the words in the text on which it already has information, but it will leave for manual treatment those words that it does not already know and those words which remain ambiguous after any disambiguation routines have been applied. This is the method which has been used to construct the *Mittelhochdeutsche Begriffsdatenbank* at the Bowling Green State University in Ohio and the Christian-Albrechts-Universität Kiel (cf. Schmidt 1991). In contrast to computer-assisted annotation, fully automatic annotation aims to assign the correct semantic fields deterministically to all known words in a text, that is, without any manual intervention and without leaving any words ambiguous.

4.8 Extending Semantic Annotation: content analysis

Up to now, we have been concerned with annotating texts using general semantic field information. However, one exception to this deserves mention and that is **content analysis**. The term 'content analysis', when used in the context of text-based research,[5] may be considered to denote

a research technique that is concerned with the classification and quantification of meanings. 'Meanings' in this instance is understood not in the usual linguistic sense, but rather in terms of the operationalization of an interpretative theory, which will probably belong to a field of scholarship other than linguistics, e.g. psychology or sociology.

Although some forms of content analysis will simply identify the occurrence of specific themes, regardless of their textual extent (words, sentences, paragraphs, etc.), others – with which we are concerned here – classify words and phrases in texts exhaustively, in a similar way to semantic annotation. These forms of content analysis are normally automatic or computer-assisted.

Manually executed forms of content analysis had been in existence since the 18th century, but a major expansion in the application of content analysis came during the Second World War, when it was applied to a significant extent – especially by the United States Government – to propaganda detection (see, for example, George 1973; Lasswell *et al.* 1949/1965). During the 1950s and 1960s, it became a very widely used methodology in many branches of the social sciences. From the late 1950s onwards, a large amount of work was devoted to automating the process of content analysis, which resulted in some very sophisticated programs for the automated tagging of the meanings of words and phrases in texts. The best known, and most widely used, of these programs was The General Inquirer (Stone *et al.* 1966; Züll *et al.* 1989).

The form of content analysis that categorizes words and phrases in running texts makes use of dictionaries (or lexica) in a similar way to general semantic annotation. However, as just suggested, the categories into which the words and phrases are classified will, in this instance, be based not on general semantic fields but rather on an analytical theory. For example, Colin Martindale (e.g. 1974) has long been involved in the content analysis of literary and other texts using a dictionary based upon Freudian psychoanalysis (the Regressive Imagery Dictionary). Here, words such as *meat*, *mud* and *naked*, which in purely linguistic terms would be classified in categories connected with, respectively, food, terrain and clothing, are instead categorized psychoanalytically as oral drive, anal drive, and sex drive. Another more mainstream example of a content analysis dictionary is the Lasswell Value Dictionary,[6] which concentrates especially on economic concepts: this contains categories such as WEALTH-PARTICIPANT (which includes words denoting trades and other economic roles), WEALTH-TRANSACTION (mainly verbs connected with financially-oriented transactions), and WEALTH-TOTAL (covering such things as income, resources, goods, services, and so on).

Content analysis has many practical applications, and these are spread

across all areas which make use of textual data. Some examples are:

- psychological profiling from texts: Hogenraad, Bestgen and Nysten (1995) looked at the content of Belgian terrorist communiqués using the Regressive Imagery Dictionary and two special dictionaries covering concepts of agression and emotion.
- longitudinal political research: Weber (1982) looked at patterns of content in British Speeches from the Throne from 1689 to 1795 (the policy speeches made at the State Opening of Parliament).
- management research: Kabanoff, Waldersee and Cohen (1995) looked at organizational values in large Australian organizations and at how the description of organizational change by members of an organization tied in with the ethos of the organization.
- medical sociology: Garwick, Detzner and Boss (1994) sought to identify the most salient themes occurring in the conversations of families about living with Alzheimer's Disease.

These are just a few examples of content analysis applications. There is a wealth of literature on other applications and on the methodology itself. A good place to start is Weber (1989). The Society for Conceptual and Content Analysis by Computer (SCACC) also publishes an occasional newsletter listing projects in content analysis as well as in general semantic field annotation and related areas.[7]

4.9 Concluding Remarks

In this chapter, we have outlined the principles and methods of annotating corpora with semantic field information. We have looked at what semantic fields are, what existing annotation systems exist, and what factors need to be taken into account when choosing or developing a semantic field system for corpus annotation. We also saw what methods can be used to assign semantic fields to words in a corpus and some of the problems which are entailed in this.

As we suggested in the introduction to the chapter, the semantic field annotation of corpora has a number of important applications: these lie not only in the fields that we have already mentioned (viz. areas of theoretical and descriptive linguistics such as semantics and discourse analysis, content analysis within the humanities and social sciences more generally, and information retrieval) but also in other areas of language engineering. One of the most important areas of application for corpora annotated with semantic field information is in devising and testing computer programs for accurate (i.e., greater than 95 per cent) automated

sense discrimination: this task is a crucial one in many areas such as the development of computer systems for message understanding and language generation but it requires large amounts of accurately annotated data from which to generate resources and on which to test ideas. We may expect to see a lot more of semantically-annotated corpora in the future.

Notes

1. For more examples along these lines, see Thomas and Wilson (1996).
2. See Chapter 16 of this volume for a discussion of EAGLES standards. Proposals are currently being made for EAGLES work on semantic annotation.
3. Part-of-speech tagging does not require such extensive lexica, since it can make use of heuristic information such as word endings (see Chapters 7 and 9). Semantic lexica do not, however, necessarily have to contain every variant form of a word if stemming and lemmatization routines are used to interface between text words and lexicon entries (cf. Rayson and Wilson 1996).
4. The most effective disambiguation method for semantic field (and related forms of) annotation appears to remain the *ad hoc* development of contextual templates for specific lexical items, as proposed by Kelly and Stone (1975), possibly with the addition of general sense-frequency weightings (cf. Wilson and Rayson 1993, Rayson and Wilson 1996).
5. It is also possible to carry out content analysis of visual data (paintings, films, advertisements) and of music, where images, objects, and musical themes, rather than words and phrases, are the items that undergo classification.
6. Used with the General Inquirer automatic content analysis software.
7. For SCACC and its newsletter, contact Prof. Klaus M. Schmidt, Department of German, Russian and East Asian Languages, Bowling Green State University, Bowling Green, OH 43403, USA. Email: schmidt@opie.bgsu.edu. Fax: (419) 372–2571.

5

Discourse Annotation: Anaphoric Relations in Corpora

ROGER GARSIDE, STEVE FLIGELSTONE
and SIMON BOTLEY

5.1 Introduction

Compared with the types of annotation discussed in the preceding chapters, 'discourse annotation' is likely to appear a particularly ill-defined concept. It is probable that a group of linguists, asked to suggest the aspects of discourse analysis they would like to be marked in a text, would arrive at very different answers. One group would be interested in units such as utterances at the upper end of grammar, where sentences are labelled and grouped according to their discourse function. Another group would want information structures to be marked in terms of such notions as 'center', 'theme' and 'rheme'. Yet another group would be interested in interactive phenomena, with particular reference to spoken language: for example, the tagging of words and expressions by their discourse role has been described by Stenström (1984). Some of these types of discoursal annotation will be briefly considered in the next chapter. This chapter focuses on yet another kind of discourse annotation, which is captured by the term **cohesion** (in the sense of Halliday and Hasan 1976) or **anaphoric relations**.

In concentrating here on cohesive relations in texts, we are making a virtue of necessity: in practice, very little discourse annotation of corpora has been so far undertaken, and the project we describe, which happens to represent cohesive relations, is the only one of which we are aware that has analysed a considerable body of text in some degree of detail (see, however, Section 5.4). However, there is also an argument that such relations are the most appropriate kind of discourse analysis to focus on, at this preliminary stage of what is bound to become a more common practice. Cohesion relations are closely tied to the other, less abstract levels of morphosyntactic, syntactic and semantic annotation. Further, cohesion adds another vital step to the process of marking, in corpus texts, those features that play an important role in text understanding or interpreta-

tion. It can be said that while semantic annotation, as described in Chapter 4, represents the meanings which words and expressions have by virtue of their intrinsic lexical content, this chapter deals with an extremely powerful and versatile capability which languages possess, of passing meaning 'sideways', from one part of a text to another. Without the ability to decode this aspect of meaning, machines will not be able to make more than partial sense of human language.

It has recently been shown at a conference on the subject (DAARC '96[1]) that pronouns, anaphora and anaphora resolution constitute an area of major interest not only to theoretical but to computational linguists. In particular, computational linguists are beginning to take seriously the use of corpora for training and testing software or algorithms in this area, just as they have been doing in other areas of natural language processing. It has been well-known for decades that one of the thorniest problems in such areas as machine-aided translation and information extraction is the resolution of anaphora: for example, finding out what third-person pronouns such as *it*, *they*, *he* and *she* refer to in a text whose meaning is to be processed. Two opposed points of view have their supporters: one says that only full access to linguistic and real-world knowledge can enable a machine to pinpoint the referents of such anaphors; the other says that we can travel a long way toward this goal by using an empirical approach, largely free of real-world knowledge and making use of such quantitative measures as the distance, in sentences, clauses, words, or characters, between anaphor and antecedent. The point about annotating a text with anaphoric relations (with or without accompanying grammatical annotation) is that it should enable the computational linguist to adjudicate scientifically between the knowledge-rich and knowledge-free approaches. Using quantitative methods and the information available in a suitably annotated corpus, it will be possible to test empirically how far one can get towards the automatic identification of antecedents, on the basis of purely formal aspects of texts.[2]

With such goals in mind, we at Lancaster were funded by IBM Yorktown Heights in 1989–91 to compile an 'anaphoric treebank' – that is, a treebank (or syntactically annotated corpus) to which were to be added annotations showing cohesive relations. This chapter describes in outline the annotation scheme we developed, and gives brief reference to the software used for the annotation, and examples of the annotation which resulted. Contrary to our worst expectations, this type of discourse annotation proved capable of being undertaken in a relatively consistent and accurate manner. Although there remained some rather intractable areas of indeterminacy (see Sections 5.3.9–12), these were scarcely more prevalent and problematic for the annotator than the areas of indeterminacy

one meets in syntactic annotation. As with grammatical tagging and syntactic annotation, we found that achieving consistency among different annotators required a detailed annotators' manual of guidelines, based on previous decisions or 'case law' (see Section 3.2).

5.2 The UCREL Discourse Annotation Scheme

A method of generating a corpus of texts with explicit marking of cohesion assumes a similar goal to that of a corpus with explicit syntactic marking: a large quantity of annotated text has ideally to be produced (perhaps several million words for the syntactic corpus, at least several hundred thousand words for the anaphoric corpus), so speed of human analysis is important, but not at the expense of accuracy and consistency in marking. The result is a notation scheme which does not attempt to mark all possible theoretically justifiable distinctions, and in fact tends to be theoretically fairly neutral, although it is influenced by the scheme described in Halliday and Hasan (1976) and by Quirk *et al.* (1985).

The remainder of this section gives a brief introduction to the annotation scheme applied at Lancaster in the project supported by IBM Yorktown Heights.[3] The following section (5.3) provides a more systematic summary and exemplification of the different annotation devices. A more detailed account of the notation, the linguistic principles behind its construction, and the guidelines used by the analysts in marking up a text are to be found in Fligelstone (1991, 1992). In Section 5.4, we will look at some other anaphoric annotation schemes which have recently come into existence, and may be regarded either as complementary or alternative to the present one.[4]

The basis of the notation is that, in a typical anaphoric link between a proform and an antecedent, the antecedent is enclosed in brackets and given an index number ('6' in the example below) which is unique within the text, and the proform is preceded by a symbol indicating an anaphoric referential link to that numbered antecedent, thus:

> (6 the married couple 6) said that <REF=6 they were happy with <REF=6 their lot.

Here the character < indicates to the human reader a preceding antecedent (i.e. the link is anaphoric rather than cataphoric), although from the computer point of view this would be adequately indicated by the coindexing of the numbers. A cataphoric link would have the character > on the symbol marking the proform. The characters REF= indicate a

referential link (as distinguished from substitute forms, ellipsis, etc.). Since proforms are nearly always one word long, a length indication (either explicit or implied by brackets round the proform) is unnecessary; in the few cases requiring a proform of more than one word, an explicit length indicator is included, as in:

(7 this week's winner 7) said <REF=7 he had rung (8 <REF=7 his wife 8) and <REF=7,8 they had spoken to <REF=7,8: 2 each other.

Here the symbol <REF=7,8: 2 indicates an anaphoric referential link from a proform two orthographic words long (*each other*); the link is to a pair of antecedents (*this week's winner* and *his wife*), and this is also indicated in the notation. It is possible to mark, with suitably placed question marks, doubt about the extent of an antecedent and uncertainty about a proform linking to a particular antecedent. A further addition to the notation allows a distinction between multiple reference and alternative reference; while the notation <REF=7,8: 2 means a reference to antecedents numbered 7 and 8, the notation <REF=7/8: 2 would mean a reference to either antecedent 7 or antecedent 8. More complicated examples are possible (though the analysts have rarely felt the need to use them); thus >REF=1,5/6,?22 would mean a cataphoric reference to 'antecedent' 1, either 5 or 6, and probably 22.

5.3 Details of the Annotation Scheme

In this section, we present and exemplify the main features of the scheme.

5.3.1 The scope of the scheme

This list shows roughly the range of relationships which are at least partially encompassed by the scheme:

- Proform reference (coreferential)
- Proform reference (substitutional)
- NP (non-pronominal) coreference [NP=noun phrase]
- *Indirect* definite NP anaphora
- Links expressible as 'inferrable *of*-complementation' of one NP by another
- Textually recoverable ellipsis
- Meta-textual reference
- NP predications (copular relationships)

Other marginal categories which may be noted are:

- Generic use of pronouns
- Non-specified cohesive ties

Although these may seem to lie outside the area of textual cohesion, it can be argued that they are important clues to tracking the linkages of sameness and similarity of meaning within a text. They are not discussed further in this chapter.

5.3.2 The elements of the notation

A system of numbered bracketing is used to mark a section or 'item' of text, and the relationship between that item and another is expressed by means of the notation on the other item. In effect, plain numbered brackets indicate antecedents, to which one or more anaphoric items may 'point'. However, two items which are numbered and bracketed identically are assumed to be coreferences. In the case of such (non-pronominal) NP-coreference no attempt is made to analyse one NP as the antecedent and one as the anaphor. (In the following example, and other examples in this chapter, the anaphoric annotation is given without the syntactic annotation which accompanies it in the treebank.)

> (1 Feodor Baumenks 1) , a former Nazi death camp guard, has asked the U.S. Supreme Court to allow <REF=1 him to retain <REF=1 his American citizenship, (2 the Hartford Courant 2) reported Monday. (2 The Newspaper 2) said (1 Federenko 1) , 72, is appealing a ruling handed down ...

A range of symbols is used to link an item to its antecedent. These symbols are made up of a number of elements which enable more or less information about the relationship to be encoded. These symbols may consist of up to 4 elements:

1. Direction
2. Type of relationship/reference
3. Identification of antecedent(s)
4. Additional features

In addition, the boundaries of the referring item are encoded, but this is not regarded as part of the symbol itself, since it does not convey any information about the nature of the relationship. Not all combinations of elements are allowed. The possible combinations are controlled by the editing program (XANADU) which acts as an interface between the analyst and the text to be annotated (see further Chapter 12). The following synopsis of the elements is followed by a description of the symbols which may be used in conjunction with each relationship 'type'.

Direction

<	anaphor (antecedent precedes anaphor)
>	cataphor (antecedent follows anaphor)[5]
?<>	non-directional or ambiguous between anaphoric and cataphoric
><	not endophoric (this is deliberately less committal than 'exophoric')

Relationship types

REF	=	central (coreferring) pronoun
SUBS	=	substitution form
ELLIP	=	ellipsis
IMP	=	indirect (or implied) anaphora
OF	=	NP with inferrable *of*-complement
MISC	=	miscellaneous cohesion
{{... ...}	=	NP predicative (copular relationship)
META	=	metatextual reference

Identification of antecedents

n	antecedent is nearest occurrence of item marked **n**
n,o,p...	multiple antecedents: **n,o,p...**
?n	antecedent is *probably* item **n**
n/o	antecedent is *either* **n** *or* **o**
n/o,p	antecedents are *either* **n** *or* **o**, *and* **p**

Semantic/Pragmatic Features

[s]	singular
[p]	plural
[g]	generic
[S]	secondary reference
[P]	primary reference
[x]	exclusive of addressee(s)
[i]	inclusive of addressee(s)

Features can, where appropriate, be bundled together (e.g.: [Ss] for a singular *you* in quoted speech). [S] and [P] are mutually exclusive, as are [s], [p] and [g], and [x] and [i]. These features can be seen as a shorthand way of representing feature/value pairs from a notional scheme:

reference(specific, generic)
number(singular,plural,no_number)
addressee(included,not_included)
level(primary,secondary)

A question mark preceding a feature indicates a degree of uncertainty. An additional feature [diff] is marked on certain problematic pronouns, as explained in Section 5.3.4.

5.3.3 Antecedents and NP coreference

Antecedents are bracketed and numbered; they need not be NPs. Items marked identically are assumed to corefer. Post-initial references will normally be NPs, but there are no restrictions on what is cited as an antecedent, except that a few conventions are adhered to:

1. For all relationships **except ellipsis** one attempts to mark a **complete constituent** as an antecedent, except that:
   - Non-restrictive postmodification of NPs is omitted.
   - Where an NP contains self-referring anaphora, a sensible 'core' is marked as the antecedent.
   - A nominal entity, though not a complete constituent, may be marked as an antecedent or as a coreference in (say) a premodifying position.
   - Complementizers such as *that* may be disregarded in clausal antecedents.

   Examples:

   (1 A man carrying a blue sports bag 1) ... ...was arrested when <REF=1 he...

   (2 The man 2) , who was carrying a blue sports bag... ...was arrested when <REF=2 he...

   (3 A man 3) carrying a gun inside <REF=3 his sports bag... ... was arrested when <REF=he...

   Six persons were treated for smoke inhalation at (4 Weld County General Hospital 4) and released, a (4 hospital 4) spokesman said.

2. For **ellipsis**, marking of the antecedent is made as precise as possible:

   You can (5 come 5) tonight unless you would prefer to <ELLIP=5 tomorrow.

A coreference with **multiple antecedents** can be indicated by bundling of numbers:

> On Wednesday, (6 the Lone Ranger 6) rode again, galloping into a Detroit sound studio along with (7 the Green Hornet 7) and (8 Sgt. Preston of the Yukon 8) ... (6,7,8 The characters 6,7,8) originated in an elegant Detroit Mansion...

An antecedent with **'uncertain boundaries'** is indicated by the attachment of query marks to the parentheses:

> ?(9 It is true that governments can influence the outcome of the competitive process in various ways, and that particular business enterprises can be owned or backed by governments 9)?, but even when <REF=9 this is allowed for, it is not states that are generally the sole...

5.3.4 Central pronouns (including demonstrative pronouns)

In the simplest case, a single anaphor refers to a single antecedent:

> (1 Federal agents 1) said <REF=1 they seized 110 pistols and revolvers at the warehouse.

No special labelling is given to so-called *mediated* reference – see Halliday and Hasan (1976: 330). In a coreference chain consisting of pronouns and non-pronominal noun phrases, the latter are bracketed and the former are marked as anaphors:

> (2 U.S. Assistant Attorney General Robert Gagnon 2), who handled the state's...
>
> (2 Gagnon 2) said later <REF=2 he approved of the penalties... and that <REF=2 he considers the case closed.

Cataphoric coreference is indicated by using a forward-pointing arrow:

> In >REF=3 his apology Friday, (3 Reynolds 3) wrote...

Multiple antecedents are shown by bundling numbers:

> On Wednesday, (4 the Lone Ranger 4) rode again, galloping into a Detroit sound studio along with (5 the Green Hornet 5) and (6 Sgt. Preston of the Yukon 6) for <REF=4,5,6 their first rehearsal since...

If a pronoun is clearly referential but its antecedent is not in the text (this may be simply because a text sample does not include the 'real' beginning of the text), it may none the less be indexed, thus allowing other co-references to be shown:

> When ><7 he came in <REF=7 he stopped and looked round. <REF=7 He shuffled <REF=7 his feet a bit...

The >< symbol should **not** be interpreted as meaning 'exophoric' or 'deictic' (i.e. referring only to extra-textual context). It is simply a device which allows features and/or indexing to be marked on an 'antecedentless' pronoun – cf. generic pronouns, below.

The addition of semantic or pragmatic **features** is appropriate for some central pronouns. Primary and secondary reference are taken to mean, respectively, reference to a/the speaker/writer or hearer/reader of the text, and reference to a character quoted in the text. Thus in a text which is actually a verbatim transcription of speech, with no narrative, virtually all the references will be primary, whereas in most texts which

are of written origin, there will be a preponderance of secondary references (though not necessarily in newspaper columns, letters, etc., which address the reader directly). Example:

">REF=8[S] I am leaving Sunday for six or seven days in England then >REF=8 I am going to California...", said (8 Eddery 8).

Note, above, that the feature is added only to the first instance of *I*. Subsequent first person coreferences are assumed to inherit the same features unless a change is indicated by the analyst. The following example from a radio broadcast illustrates how this can happen:

(9 The minister 9) said at lunchtime "<REF=9[S] I have absolutely no comment to make", but (9 the minister 9) is in the studio with me now. (9 Minister 9), are <REF=9[P] you now in a position to add anything to the statement <REF=9 you made earlier today?

A feature [diff] is sometimes used to indicate a pronoun which is felt to be anaphoric, but whose antecedent is for some reason **difficult** to identify. This is particularly common with first person plural pronouns and with *it*. Appropriately, the notation closely resembles that used for the 'antecedentless' cases just considered. The difference between the categories is that in one case, there is simply no antecedent in the text, whereas in the other, there is felt to be one, but one which eludes identification! In both cases, the '><' marker is used, and a new index number generated, enabling subsequent coreferences to be resolved elegantly.

Compound coreferential proforms are marked with an extra field indicating the number of words in the expression:

<n: 2 one another
<n: 2 each other

5.3.5 Substitute forms

As well as standard substitute forms such as *one, ones, others*, etc. (see Halliday and Hasan 1976: 88–141; Quirk *et al.* 1985: 865–83), this category has been defined as including the following:

1. Pro-verbs such as *do, do so, do (just) that, do likewise, do it*:

   Asked if he would (1 appeal 1), Smith would only say that he may <SUBS=1 do.

2. All (anaphoric) numerals when in NP head position (i.e. numerals are not analysed as NPs with elided heads):

The benefit paid for her will be lost if her income exceeds (2 thirty pounds 2) instead of the present <SUBS=2 forty-five.

3. All ambiguous pronoun/determiners functioning as NP head are assumed to be pronouns. Again, this results in a variety of words cast in the role of 'substitute forms' (that is, if they are not classed as central coreferential pronouns, or as non-anaphoric, etc.):

   (3 Three army parachutists 3) dropped from a plane. <SUBS=3 One carried the flag of Greece. <SUBS=3 Another carried the flag with the five rings.

   After each couple has skated, the judges post (4 two sets of scores 4). A maximum score in <SUBS=4 each is 6.0.

4. The locative and temporal adverbs *here*, *there* and *then* when their reference occurs explicitly in the text. If a prepositional phrase is available, the whole constituent is marked as the antecedent:

   Delegates were expecting to see the Rev. Jesse Jackson (5 at the convention 5), and were disappointed to discover that he was not <SUBS=5 there.

   However, there is frequently no prepositional phrase:

   Quite late in life he wrote (6 his famous novel, 'Candide' 6). <SUBS=6 Here he rejected the Leibnizian view...

   ?(7 Normally 7)? visitors to the State Department require credentials and even <SUBS=7 then they have to pass through metal detectors.

5. *So* in expressions like *I think so* and *if so*:

   (8 Has the Department of Transport begun a review of its plans for expansion at Mount Hope Airport 8) and, if <SUBS=8 so...

As in the case of coreferential proforms (Section 5.3.4), **compound substitute forms** are given an extra field after the index number, which indicates the number of words in the expression:

I strongly encourage you (9 to repent and live the gospel standards, to prepare yourself for rebaptism 9), and it is my prayer that you may <SUBS=9: 2 do so...

5.3.6 Ellipsis

We mark ellipsis in contexts such as the following:

- Headless NPs (but see Sections 5.3.5 (2) and 5.3.5 (3))
- Stranded infinitive markers and auxiliaries
- Answers to questions
- Gapping

Only ellipsis which can be termed 'precisely recoverable' is marked. Even where 'recoverability' may be felt to exist, constructions defined as co-ordination (see Quirk *et al.* 1985: 941–5) are not counted as ellipsis. For example, *Susan and her husband came late*, is considered to be not an elliptical form of *Susan came late and her husband came late*, but a single clause with a coordinated NP a subject. Further, a number of constructions which are arguably ellipsis are in fact marked as substitution (see Section 5.3.5). The pro-verb *do* is treated as a substitute form only when it is a main verb. Otherwise it is treated like any other auxiliary, which may be followed by ellipsis:

By offering the (1 motion 1) as his own <ELLIP=1 , Rep. Richard Bolling was able...

He was (2 upset 2) not so much about the decision to fight as he was <ELLIP=2 about the decision being made by the clerk's office.

What's the good of everyone (3 blocking out 3) if only four do <ELLIP=3 and one doesn't ELLIP=3 and the other team end up with the rebound?

5.3.7 Implied antecedents

Since implied or indirect anaphora is a somewhat indeterminate area of cohesion, we have had to impose fairly strict guidelines. This label is applied only to **definite NPs** which link anaphorically to an item of text from which an 'implied antecedent' can be recovered. Furthermore it must be possible to construct such an implied antecedent by forming a prepositional link between the second NP and the first. (If, however, the only preposition which could plausibly be used to express the relationship between NP2 and NP1 is *of*, the item is labelled as described in the next section.)

He groped over the surface of (1 the rock 1), feeling it, almost hugging it, in the desperate need to find <IMP=1(2 the entrance 2).

He put on (3 his goggles 3), fitted them tight, then tested <IMP=3(4 the vacuum 4).

There was no apparent clan presence at (5 the march 5). About 5,000 marchers had been expected, but one (5 march 5) organiser said <IMP=5(6 the turnout 6) had apparently been limited by cold weather...

It will be noted that in these examples the second item is indexed in its own right, so that it can be used as an antecedent, as well as being linked to the source of recovery of the implied antecedent. We have designed our editing program XANADU to do this automatically, although in some cases this indexing will be unnecessary.

5.3.8 Inferrable of-complementation

The criteria for inclusion in this category are different from those for IMP=, although there is some overlap. There is no need for items in this category to be definite. Given two NPs X and Y, it must be possible to 'conjure up' a natural-sounding expression 'Y of X'. In the case of definite NPs, if *of* is merely one of several prepositions which would fit, the category IMP= is marked, but if *of* is the only plausible preposition, then the category OF= is marked. This category typically includes a variety of quantitative expressions and noun phrases whose heads often occur with PP-complementation:

> As she described (1 her daughter 1) to me, I imagined an attractive woman in her late twenties. But when she showed me <OF=1(2 a picture 2) I was dumbstruck.

> He took the lead on the 65th lap of (3 the 80-lap race 3) and cruised to victory under the yellow caution flag after a car spun out on <OF=3(4 the 77th lap 4).

5.3.9 Miscellaneous cohesion

This category may be used for any cohesive relationship between X and Y (where Y is a noun phrase) not covered by the other conventions of the scheme. Its scope has not been defined beyond this in the Lancaster scheme. The function of this label is to act not so much as a dustbin as a pressure valve, making it less likely that frustrated annotators will 'force' relationships into categories whose criteria they do not meet. However, MISC= also captures some interesting cohesive phenomena, and it may also turn out that some of the things typically analysed as MISC= can be categorized and given explicit labels at a later stage of refinement. Up to now, however, we have resisted adding further complexity to the scheme. Here is an example of where the label has been used:

> He will not discuss (3 prices 3). But <MISC=3(4 the tab 4) comes to $4000 for a seven-point elk.

5.3.10 Metatextual references

The analysis of metatextual references is limited to an indication of direction. There are four possible markers:

<META	anaphoric reference (e.g. *above left*)
>META	cataphoric reference (e.g. *overleaf*)
?<>META	ambiguous or non-directional reference (e.g. *this chapter*)
><META	exophoric reference (e.g. *elsewhere*)

A multi-word item is denoted by use of a number field (cf. substitution, Section 5.3.5):

In <META:4 the last few sections, we...

5.3.11 NP predications (copular relationship)

NP predications expressing a copular relationship are indicated. The subject of the predication is simply marked as if it were an antecedent. The predication itself is given a directionality which is syntax-based. Copular relationships have been recognized in four syntactic contexts:

1. Relation between subject and complement of a copular verb (e.g. *be*):

   (1 all of the inmates in the program 1) are {{first offenders 1}.

2. Relation between two noun phrases in apposition:

   (2 Wallace's roommate 2), {{2 Jim Stutzman 2} (full apposition)
   {5 manicurist 5}} (5 Merle Conrad 5) (partial apposition)

3. Relation between direct object and object complement:

   The victory by (3 the Dominican Republic 3) gave (3 the host team 3) a 3–0 record and made <REF=3 them {{3 a solid favorite to win the four-team competition 3}.

4. Copular relation between two noun phrases and signalled by *as*:

   Police described (5 the incident 5) as {{5 the worst mass slaying in the city's history 5}.

5.3.12 Additional aspects of the annotation scheme

1. *Cohesion barriers* The symbol (0) is inserted when a natural or virtual text end is reached, for example between two newspaper reports. This enables indexing to be recommenced from '(1)', since otherwise some large numbers might be generated. It is also used to prevent spurious linkage being marked – for example a 'cohesive' tie between two unconnected mentions of *America* in two adjoining texts.
2. *Commenting* The XANADU editor allows the analyst to insert comments at any point in the text. This was a useful communication device during the development and learning phrases of the scheme. It is also a means by which one can 'flag' items in the text for later retrieval. It is even open to the analyst to use the commenting facility to devise a personal annotation scheme with which to augment the primary conventions.

5.4 Some Other Anaphoric Annotation Schemes

It can be said that discourse annotation involving anaphora is a relatively new area, with little work actually having been carried out up to the present. Such annotation schemes tend to be developed for particular corpora, such as those used in doctoral research (see Botley's scheme below), while others have a wider provenance. We will briefly describe three schemes identified in a review of current research.

5.4.1 De Rocha's annotation scheme

In a scheme developed by Marco de Rocha (de Rocha 1996), spoken corpus texts[6] in English and Portuguese are segmented and annotated according to the topic structure of the texts analysed. The approach reflects the widely accepted view in discourse analysis and text linguistics that the topic of the discourse[7] tends to be the preferred antecedent for a given anaphoric expression. Thus de Rocha's annotation is aimed at exploring the complex relationships between anaphora and discourse topic.

First, de Rocha establishes, for each discourse fragment under analysis, a global topic, or **discourse topic**. The discourse topic can be valid throughout a whole text, or may change at different points, in which case a new discourse topic will be established and annotated. The discourse topic is annotated above the text fragment as a noun phrase within asterisks. The next step is to divide the text into discourse segments according to topic continuity. This is done by assigning a **segment topic**, which is valid throughout a given segment of discourse. Whenever the local topic changes, a new segment topic is assigned, and an appropriate annotation is manually inserted into the text. Segment topics are annotated using the letter **s**, followed by an index number similar to those assigned by the UCREL discourse scheme in Sections 5.2–3 above. Segments where further local topic shift occurs are further subdivided into subsegments, with their own annotation, consisting of the string **ss** followed by an index number as with segment annotations. Also, topics which have been dropped, but have been re-introduced in the conversation, are also marked by adding the letter **r** to the **s** or **ss** annotations for discourse segments.

As well as the above segment and subsegment annotations, de Rocha's scheme allows for discourse segments to be annotated according to the discourse function they serve, for instance 'introducing the discourse topic' is annotated intro_dt. Also, each annotation string contains a short phrase describing the current topic for the segment or subsegment under analysis. The final stage of de Rocha's analytical framework is the annotation of

each case of anaphora taking place within the discourse segments identified. Four properties of anaphora are specified:

1. **Type of anaphora**, such as 'subject pronoun' or 'full noun phrase', each type having its own tag.
2. **Type of antecedent**, defined as either implicit or explicit, each of which is tagged separately in the annotation.
3. **Topicality status of the antecedent**, in other words whether the antecedent is the discourse topic, segment topic or subsegment topic.
4. **Processing slot**, by which cases of anaphora can be classified according to the type of knowledge used in processing them, such a syntactic, collocational or discourse knowledge.

To illustrate the system, here is a piece of text, fully annotated in accordance with de Rocha's scheme:

```
*s1 intro_dt 'B's thesis'
004.0005 A how's the thesis going ,
**              (DA) FNP
***             im_1; dt; SK;
005.0006 B uh I'm typing it up now .
**                    OP
***             ex_1; dt; FtOp;
005.0007 B typing up the final copy
**              (DA) FNP
***             im_2; thel; WK;
```

Evaluation of de Rocha's annotation scheme

De Rocha's scheme has a number of innovations. First, it goes beyond annotating anaphoric relations in texts, and attempts to encode information about the relation between anaphora and topicality. This goes some way towards providing annotated corpora that can be used in studies linking anaphora to discourse structure. Secondly, rather than simply identifying anaphors and antecedents, it classifies them according to criteria which are more detailed than the framework of Halliday and Hasan at the core of the UCREL scheme. Thirdly, de Rocha's scheme has been developed for use with spoken dialogues in more than one language, which introduces the extra analytical dimension of cross-linguistic comparisons. Finally, de Rocha introduces information concerning the kind of knowledge used in processing anaphors, which is not included in other schemes, and should be valuable for studies which marry corpus-based descriptions to a knowledge-based approach to anaphor resolution.

One disadvantage of de Rocha's scheme is that it does not use a widely-accepted text encoding format, and the opacity of its symbols ignores the requirement of 'user-friendliness' that may become more important in modern corpus-based research.

5.4.2 Gaizauskas and Humphries' annotation scheme

Gaizauskas and Humphries (forthcoming) take another desirable approach to labelling, which is to use SGML (Standard Generalized Markup Language) tags to annotate anaphoric expressions in texts used in a coreference resolution task. Gaizauskas and Humphries describe their basic SGML tagging framework as follows:

Given an antecedent A and an anaphor B, where both A and B are strings in the text, the basic coreference annotation has the form:

```
<COREF ID="100"> A </COREF>
<COREF ID="101" TYPE=IDENT REF="100"> B </COREF>
```

So, for example, *Galactic Enterprises said it would build a new space station before the year 2016* would be marked up as (Gaizauskas and Humphries forthcoming):

```
<COREF ID="100"> Galactic Enterprises </COREF> said <COREF ID="101"
TYPE=IDENT REF="100"> it </COREF> would build a new space station before
the year 2016
```

The various SGML attributes in the above example are:

- ID functions as a unique and arbitrary identifier for each string in a coreference relation
- REF identifies which string is coreferential with the current string
- TYPE indicates the kind of relationship between anaphor and antecedent
- IDENT indicates identity between anaphor and antecedent, for instance with lexical repetition.

Also, there is a STATUS attribute, which takes the value OPT for cases where an analyst indicates genuine uncertainty concerning the antecedent of an anaphor.

Evaluation of Gaizauskas and Humphries' system

The system's main advantage is being in a widely-recognized text interchange format. However, it allows only a small subset of anaphoric relations to be marked. The scheme was devised for use in a rigidly restricted automatic anaphora resolution task where the success of each annotation had to be measured precisely. It was not developed for use on a large-scale corpus project, like the other annotation schemes described in this chapter. Despite this, the SGML framework does provide a useful model as a starting point for other schemes which might convert to SGML in the future.

5.4.3 Botley's annotation scheme

The final annotation scheme to be described here was developed by Botley (Botley 1996) and, like that of de Rocha, it attempts to classify anaphoric expressions according to various externally valid criteria. Botley's scheme was developed to describe the different ways in which demonstrative expressions function anaphorically in written and spoken corpus texts. Essentially, he classifies demonstrative anaphors according to five distinctive features, each of which can have one of a series of values. The distinctive features, with their possible values, are as follows:

- **Recoverability of antecedent** (the extent to which the antecedent is a recoverable surface string in the text)
  - D = directly recoverable antecedent
  - I = indirectly recoverable antecedent (cf. implied antecedents, Section 5.3.7)
  - N = no recoverable antecedent
  - 0 = not applicable (e.g. exophoric)
- **Direction of reference**
  - A = anaphoric
  - C = cataphoric
  - 0 = not applicable (e.g. exophoric or deictic)
- **Phoric type** (essentially derived from Halliday and Hasan)
  - S = substitutional
  - R = referential
  - 0 = not applicable (i.e. non-phoric)
- **Syntactic function of anaphor**
  - M = modifier function
  - H = head function
  - A = adverbial function
  - 0 = not applicable (syntactic function irrelevant because antecedent not present)
- **Antecedent type**
  - P = propositional / factual antecedent
  - C = clausal antecedent
  - N = nominal antecedent
  - J = adjectival antecedent
  - 0 = not applicable (no antecedent)

Each case of demonstrative anaphora in a corpus has been manually tagged with a five-character string consisting of one of the above values for each of the five features identified. Here is a corpus example containing a demonstrative phrase:

"Our offense is designed to shoot lay-ups. If we can't carry on **this offense**, we find ourselves sitting on the bench." (A005: 7–8)

This case is tagged DARMN, because its antecedent is Directly recoverable, it is Anaphoric, it is Referential in phoric type, is a Modifier in syntactic function and has a Nominal antecedent type.

Evaluation of Botley's annotation scheme

Although it does not mark the position and extent of the antecedent, Botley's scheme, like de Rocha's, has the advantage of being able to mark a great deal more information about anaphoric phenomena in the text than the UCREL scheme at present can. Also, it is relatively straightforward to derive statistics concerning frequency of occurrence of particular demonstrative features using the Botley scheme, from which sophisticated statistical modelling such as logistic modelling (see Leech *et al.* 1994) can be carried out. Schemes which classify antecedents according to directness or indirectness of recoverability (Botley) or explicitness versus implicitness (de Rocha) are valuable and sensitive tools to help analysts to derive richer descriptions of particular anaphoric features in a corpus. The only question of doubt is whether these schemes, which were devised by individual researchers primarily for their own work, would enable sufficient accuracy and consistency of annotation to be achieved, or whether some of the features they identify are too abstract or uncertain to form the basis for a commonly agreed annotation scheme.

5.5 Conclusion

From the comparison of the different annotation schemes in Sections 5.3 and 5.4, it appears that discourse annotation representing cohesion is at a fairly immature stage of development, where there is virtually no sense of a standard set of practices to be followed. Partly, the impression that everyone is 'doing their own thing' comes from the fact that the four annotation schemes have varying purposes: de Rocha's and Botley's are essentially for the annotator's individual or local use as a research tool; Gaizauskas and Humphries are aiming for a common interchange format across different research centres; whereas the UCREL scheme is somewhere between these extremes. Also, more optimistically, it seems that these schemes do not conflict with one another (except in superficial ways, in terms of notation or layout), but rather complement one another: thus, de Rocha and Botley add further dimensions of information to those already present in the UCREL scheme. Perhaps, then, we are justified in

characterizing the UCREL scheme as somewhat analogous to the skeleton parsing scheme described in Section 3.2: it is a basic core of widely agreed annotations, which enables annotators to achieve a rather high degree of consistency, and to which other annotators may add more abstract or complex levels of annotation as more sophisticated needs and applications arise. The lesson suggested by Gaizauskas and Humphries is that the UCREL scheme, if it to become more widely used, needs to be readily translatable into a recognized interchange format based on SGML.

Notes

1. The proceedings of the DAARC '96 conference are being published in Botley and McEnery (forthcoming). See also Botley *et al.* (1996a).
2. Mitkov's work (e.g. Mitkov 1995, Mitkov forthcoming) and a survey of others' research show how corpora are beginning to be used in the testing of anaphoric resolution algorithms. However, the use of anaphorically – and grammatically – annotated corpora in testing such software has as yet remained undeveloped, although it may be seen as strictly parallel to the use of parsed corpora for training and testing parsers (see Section 3.1, Chapter 11).
3. We acknowledge the support and advice in 1989–91 of the Continuous Speech research team at the Thomas J. Watson Research Center, IBM Yorktown Heights, led by Fred Jelinek.
4. The remainder of this chapter is substantially based on Garside (1993) (in Section 5.2) and Fligelstone (1991) (in Section 5.3). Section 5.4 is contributed by Botley.
5. We use the term antecedent (in spite of its etymology) for coreferential constituents which follow, as well as precede, the proform; that is, for cataphora as well as anaphora.
6. de Rocha used extracts from the London-Lund Corpus for his English data.
7. 'Topic' is known by various terms in the literature, for instance 'focus' (Sidner 1986) or 'center' (Mitkov 1994).

6

Further Levels of Annotation

GEOFFREY LEECH, TONY McENERY
and MARTIN WYNNE

As the last of the chapters on levels of corpus annotation, this chapter will give a brief overview of three types of corpus annotation not covered in the preceding four chapters, namely prosodic, pragmatic and stylistic annotation. For various reasons, these cannot be regarded as 'mainstream' levels of annotation, yet they have considerable interest in their own right, and illustrate the kinds of direction in which corpus annotation may move in the future.

6.1 Prosodic Annotation

When prosodic annotation was discussed in Section 1.4.2, it was pointed out that there is some doubt, indeed, as to whether prosodic annotation is a kind of annotation, or whether it should not be regarded as part of the primary data. After all, the prosodic symbols are representations of part of a spoken transcription, indicating the way in which a piece of spoken language was uttered. On the other hand, they also resemble annotations at other levels, since they are a basically a linguist's or phonetician's interpretation, through linguistic analysis, of the primary signal of speech itself. We may argue, in fact, that the text itself is represented not by the transcription but electronically by the recording. Nowadays this is in principle capable of being digitized and cross-referred to by anyone working with the prosodic or other transcribed rendering of the text. However, it would be futile to continue this argument further. For our purposes, prosodic labelling[1] is being included in this book, since it is an area where a considerable amount of interesting and productive work has been done.

6.1.1 Two prosodically analysed corpora of English: the LLC and the SEC

Our main discussion will centre on two spoken British English corpora which have been used as a basis for research: the London-Lund Corpus (LLC) (begun in 1975) and the Lancaster/IBM Spoken English Corpus (SEC) (begun in 1984). Both have been made widely available for research (through the Norwegian Computing Centre for the Humanities, Bergen) in their electronic form, and have also, as a matter of interest, been published in book form (Svartvik and Quirk 1980; Knowles, Williams and Taylor 1996).[2] Although our main focus is on prosodic labelling, rather than on the corpora themselves, Table 6.1 points to some of the main differences between them.

From Table 6.1 it is easy to see where the relative strengths and weaknesses of the two corpora lie. These stem from the different purposes for which they were built. The LLC originated in the pre-computer-corpus days of 1960, and was collected and transcribed as part of the non-electronic Survey of English Usage corpus assembled by Randolph Quirk at London (Quirk 1960). Later, Jan Svartvik of Lund initiated a process of conversion to electronic form, entailing a considerable simplification of the transcription system. The recordings were made in relatively early days, and the primary purpose of the corpus was for linguistic research. The main purpose, on the other hand, for which the SEC was built was for advances in speech technology – in particular, for developing high-

Table 6.1 Comparison of two prosodically labelled corpora[a]

	London-Lund Corpus	Lancaster/IBM Spoken English Corpus
Size	Large: 435,000 words (later 500,000 words)	Small: 52,639 words
Type of interaction	Monologue and dialogue	Mostly monologue
Degree of spontaneity	Varied (contains a great deal of spontaneous conversation)	Mostly unspontaneous, scripted
Levels of annotation	Prosodic	Prosodic, orthographic, grammatical tagging, skeleton parsing
Detail of prosodic labelling	Fairly detailed	Not so detailed
Availability of recordings	Not generally available	Generally available for academic use

[a]The differences between the LLC and the SEC are explained in much more detail by Williams (1996a).

quality speech synthesis. Hence, for this corpus, it was important that the quality of recordings was very high, and that the speech should be highly articulated formal speech, rather than speech recorded in informal conversational situations, where background noises, non-fluency phenomena, etc. would have vitiated its purpose. These different objectives have also been evident in the focus of the publications reporting research making use of each corpus: in particular, it is valuable to consult the two books by Svartvik (1990) and Knowles, Wichmann and Alderson (1996). It cannot be denied that the LLC, as is true of many corpora, has been used for a multitude of research purposes unlikely to have been foreseen by its originators[3] – for example, apart from obvious things such as the

Table 6.2 London-Lund Corpus: main symbols of the prosodic transcription. (Adapted from Svartvik and Quirk 1980: 22.)

Nucleus	YES	Fall
	YES	Rise
	YES	Level
	YES	(Rise-) fall-rise
	YES	(Fall-) rise-fall
	YES YES	Fall-plus-rise
	YES YES	Rise-plus-fall
Booster	▷yes	Continuance
	△ yes	Higher than preceding syllable
	△ yes	Higher than preceding pitch-prominent syllable
	△ yes	Very high
Stress	'yes	Normal
	•yes	Heavy
Tone Unit	‖	End of tone unit (TU)
	‖	Onset
	{yes}	Subordinate TU

Table 6.3 Lancaster/IBM Spoken English Corpus: Symbols of prosodic transcription. (From Knowles, Williams and Taylor 1996: 3.)

|	Minor tone-group boundary	_	Low level
||	Major tone-group boundary	ˇ	High fall rise
^	Caret	ˇ	Low fall rise
\	High fall	^	Rise fall
\	Low fall	·	Stressed but unaccented
/	High rise	↑	Up arrow
/	Low rise	↓	Down arrow
¯	High level		

grammatical composition of speech, the use of discourse markers and the occurrence of non-fluency phenomena, less obvious things like automatic speech segmentation, statistical models of language variation, joking, and information flow in conversation. On the other hand, although the SEC has also been widely used for linguistic research, its more specialist applications have been in speech science and speech technology – the use of technology to explore the nature of speech, and to synthesize or interpret spoken language.

Turning now to the prosodic labelling, both corpora belong broadly to the British tradition of representing intonation by **tonetic stress** marking (Kingdon 1958): i.e. representing nuclei, boosters and other stressed syllables by symbols preceding the syllables concerned. The main

Example 6.1 A brief sample from the London-Lund Corpus. (From Svartvik and Quirk 1980: 174.)

A 58 we ‖actually 'ran out of TE\A■ 59 ‖one WE\EKEND you KNO/W■ 60 ‖once on a △SU\NDAY■ — 61 ‖must have 'made a lot more than we 'usually △DO\ or■ 62 ‖SO/METHING■ 63 or [z] ‖all △I/ 'know■ 64 is ‖when she 'went to △see Mr △GRE\Y■ 65 that ‖horrible 'weekend I was △left on my △O\WN■ ·

a 66 [m] ·

> **A** 67 and ‖Milly sent △some of these △BA\GS 'back■ ·

This is the 'reader-friendly' format of the LLC prosodic labelling as published in Svartvik and Quirk (1980). A less attractive, but more convenient format, is the ASCII version illustrated in Box 1.3, Example 1a in Chapter 1.

Example 6.2 A brief example from the SEC. (From Knowles, Williams and Taylor 1996: 74.)

> /Nemo | the _killer ,whale | who'd _grown | too ¯big | for his ,pool | on _Clacton ,pier | has a¯rrived ,safely | at his ¯new ,home | in _Windsor sa·fari \park || but the ˇjourney | was _not wi·thout \mishaps || ˇNemo^ ·weighs | ¯one and a ·half tons | and he ¯nearly _proved | ¯more than a \match | for a \crane | ¯brought in to \lift him | from the \pool || ¯our re_porter | ¯Peter _Burden | has been ↓ ¯following e\vents ||

features of the two systems of prosodic transcription are represented in Tables 6.2 and 6.3 and in Examples 6.1 and 6.2.

6.1.2 Other issues of prosodic labelling

Before we leave the subject of prosodic annotation, there are a number of issues to be mentioned:

1. A distinction may be made between a **spoken language corpus** and a **speech corpus**. 'Speech corpus' is the term speech scientists sometimes use to refer to databases of 'laboratory' samples of speech, typically consisting of sets of words or sets of sentences out of context, spoken by a range of speakers of different dialects, different genders, etc. For English, the best example of this is the DARPA-TIMIT database developed in the US (see Williams 1996a: 11). Corpus linguists would normally take a stand on avoiding the use of the term 'corpus' for such databases, because they do not represent samples of naturally-occurring continuous speech: a corpus is normally taken to be natural and authentic. Nevertheless such speech corpora are key resources for speech technology – e.g. for developing more natural-sounding speech synthesis at a 'micro' level. A corpus of continuous spoken discourse such as the SEC is particularly useful for the 'macro' level of improving our knowledge of how natural intonation behaves over longer stretches of discourse – the spoken equivalent of sentences and paragraphs.
2. There is a growing number of corpora of continuous spoken language or discourse which do not have prosodic labelling (for example, the spoken part of the BNC consisting of 10 million words). In fact, making a prosodically annotated corpus such as the LLC or the SEC is an extremely time-consuming business, and also requires the commitment of one (or preferably more than one) highly-trained phonetician over long periods of years. It is therefore no wonder that such corpora are rare. On the other hand, it can be reasonably argued that **prosody**

(stress, intonation and related phenomena) is as essential to the representation of spoken language as punctuation is to the representation of written language. The transcription of spoken language into ordinary orthography, which has been the simplistic type of transcription used for the BNC, the Bank of English and all of the larger spoken corpora, is a pseudo-procedure the only excuse for which is that it would be prohibitively expensive to attempt anything else. We have to admit that such orthographically-transcribed spoken English corpora can be used only for limited types of research: for example, research into spoken English vocabulary or (less satisfactorily) grammar.

3. There are nevertheless some forward-looking developments in prosodic labelling that should be mentioned. One is the compilation of a Corpus of Spoken American English, which is being transcribed with a well-thought-out set of prosodic conventions, particularly suitable to the use of spoken corpora in discourse analysis (Du Bois *et al.* 1990, Du Bois and Schuetze-Coburn 1993). Another, coming from a very different academic direction, is an American computerized system (TOBI) for producing prosodically labelled text, at a much greater speed than can be achieved by the intensive manual transcription methods employed for LLC and SEC. The TOBI system is based on the American tradition of intonation analysis descending from Trager and Smith (1951) through Pierrehumbert (1980) (see Roach and Arnfield 1995: 155–60, Williams 1996b). The differences between American and British prosodic mark-up traditions here have much to do not only with the intellectual styles of phoneticians in the two countries, but also perhaps with the dialectal differences in intonation between American and British English.

Finally, mention should be made of a movement towards the development of an SGML-based standardized mark-up for spoken discourse, including stress and intonation. This is one of the more successful areas of text encoding which have been developed under the Text Encoding Initiative (see Johansson 1995).

6.2 Pragmatic Annotation

Since the 1970s, the field of **pragmatics** has become increasingly significant as a branch of linguistics, and more recently there has been a fairly sudden realization of the relevance of pragmatics to natural language processing and language engineering. As the preceding section on prosodic labelling also has strong affinities to pragmatics, it is easy to see that the

annotation of pragmatic features could be a useful adjunct to prosodic and syntactic mark-up in a corpus of conversational data. In this section, we will study in detail one approach to the mark-up of speech acts in the **verbal response mode** (VRM) system of Stiles (1992). We will concentrate on the work of Stiles since, although it is comparatively less well known than the work of other analysts, notably Sinclair and Coulthard (1975), it represents contemporary, corpus-based work which makes significant claims within one domain of discourse (that of doctor–patient interaction).

Stiles (1992) produced a taxonomy of speech act types, which he called verbal response modes. Each speech act can be typified as having one of eight functions, summarized in Table 6.4. However, the full range of VRMs is expressed by a combination of two VRMs: one to show what Stiles terms 'literal meaning' and another to show what Stiles terms 'pragmatic meaning'. This distinction of pragmatic meaning can best be understood if it is noted that it is simply a reformulation of Grice's concept of occasion meaning. Stiles's system allows 64 (8×8) types of speech acts. These are summarized in Table 6.5 (overleaf). Table 6.4 may be used as a key for interpreting Table 6.5.

Stiles' work is of interest in that in producing this classification he subsumed earlier partial classifications of conversational acts, such as those of Kiesler (1973), Beattie (1983), Viney (1983), and Russell (1988). This was a great advantage, since these authors had built a set of speech act classifications that were not complementary, but partially redundant, as noted by Elliot *et al.* (1987). Stiles used his VRMs to encode a two-million-word corpus of conversations, typifying each utterance with a speech act type.[4]

Using his corpus, Stiles discovered a grammar of discourse for this

Table 6.4 An eight-part analysis of speech act types

Disclosure (D)	Reveals thoughts, feelings, perceptions or intentions
Edification (E)	States objective information
Advisement (A)	Attempts to guide behaviour; suggestions, commands, permission, prohibition
Confirmation (C)	Compares speaker's experience with others; agreement, disagreement, shared experience
Question (Q)	Requests information or guidance
Acknowledgement (K)	Conveys receipt or receptiveness to other's communication; simple acceptance; salutations
Interpretation (I)	Explains or labels the other; judgements or evaluations of the other's experience/behaviour
Reflection (R)	Puts other's experience into words; repetitions, restatements or clarifications

Table 6.5 A 64-part analysis of speech act types including both literal and pragmatic meaning

		Pragmatic Meaning							
		D	E	A	C	Q	K	I	R
Literal Meaning	D	DD	DE	DA	DC	DQ	DK	DI	DR
	E	ED	EE	EA	EC	EQ	EK	EI	ER
	A	AD	AE	AA	AC	AQ	AK	AI	AR
	C	CD	CE	CA	CC	CQ	CK	CI	CR
	Q	QD	QE	QA	QC	QQ	QK	QI	QR
	K	KD	KE	KA	KC	KQ	KK	KI	KR
	I	ID	IE	IA	IC	IQ	IK	II	IR
	R	RD	RE	RA	RC	RQ	RK	RI	RR

particular type of interaction. The system of coding he had developed showed that the type of utterance initiating an interchange and the context of that interchange combined to structure the interchange through a series of fairly predictable[5] speech acts.[6]

In the quantitative analysis of a corpus of doctor–patient interactions, Stiles discovered a distinct set of verbal interactions, exemplified in Table 6.6. Example 6.3 is an exposition exchange, which should help to illustrate the scheme of analysis. The exchange is taken from Stiles (1992: 54). Within this mode, the doctor is limited to producing utterances that have both a literal and pragmatic meaning of acknowledgement. The patient, while having a wide range of VRMs available, is in fact limited to a subset of five speech act types (from the sixty-four possible within Stiles' system) mainly centred upon edification. It is the context of the discourse that leads to the particular interpretation of the speech act.

For instance, there is nothing inherently DD about *I have the headaches to the point where I want to throw up with them.* Given different contexts it may have different meanings. For instance, if the hearer also has the headache there may be an element of acknowledgement to the utterance (DK). As another example, if the other person has the headaches too, and the hearer is stating that they 'throw up' as an expression of how bad the headaches are, the speech act may ostensibly be a disclosure, but act as a question to discover whether the other person throws up too (DQ). However,

Table 6.6 Three examples from the eight classes of interaction Stiles discovered in doctor–patient interaction.

Exposition Dr: KK Patient: DD, DE, EE, ED, QQ Context: Throughout, but especially in history Description: Patients describe their illness and circumstances in their own words
Closed Question Dr: QQ, RQ, EQ Patient: KE, KD Context: History and conclusion Description: Physicians ask specific questions; patients give brief answers
Checking Dr: RR, ER Patient: KC Context: Mainly in history Description: Physicians repeat or summarize information given by patients

these responses seem excluded by the interaction type: DQ and DK responses do not occur within exposition exchanges. So it is the context, both in terms of preceding utterances and the environment of the interaction, that determines the pragmatic, as well as to a lesser extent the literal, interpretation of the meaning.

Stiles' work is important as it gives empirical, corpus-based evidence

Example 6.3

Patient:	I have the headaches to the point where I want to throw up with them (DD)
Dr:	Mm-hm (KK)
Patient:	Then I have to go to bed with them (DD) And then I'll go to sleep for awhile (DE) and maybe they will ease off. (ED) But after a while, they'll come right back again. (ED)
Dr:	Mm-hm (KK)
Patient:	And my blood pressure last Saturday started dropping. (EE) And that's when they admitted me to hospital. (EE) And I've been in hospital all week. (DE) Well, (KK) I've had headaches all week in hospital. (DD)
Dr:	Mm-hm. (KK)

for the existence of a discourse grammar. A knowledge of discourse conventions and structures is important, as it may determine pragmatic inferences for the resolution of uncertain references, as suggested in McEnery (1995, 1997). Hence the development of such speech act annotated corpora is of potential importance to both discourse linguists and computational linguists.

It must be admitted that there is no clear-cut distinction between 'pragmatic annotation' as exemplified in Stiles' work, and 'discourse annotation' as discussed in Section 5.1. Arguably, pragmatics develops into discourse analysis whenever the pragmatician changes from examining the way discourse meaning relates to a particular context to how discourse meaning evolves as the context of discussion evolves (see Thomas 1996: 201–3). Similarly, as we move from pragmatic annotation to stylistic annotation in the next section, we are conscious of a change of emphasis (stylistics being particularly associated with literary texts) rather than a change of basic discipline.

6.3 Stylistic Annotation

To demonstrate methods of annotation in the stylistic area of research, we will present a case study in stylistic tagging, drawing attention to problems specific, or especially relevant, to the annotation of stylistic features. The aspect of style under investigation here is the linguistic representation of people's thoughts and speech, often known as **speech and thought presentation**, and hereafter referred to as S&TP. The data are the product of the work of a group of literary stylisticians and corpus linguists in the Department of Linguistics and Modern English Language at Lancaster.[7] This started with a pilot S&TP corpus (described and analysed in Short *et al.* 1996 and Semino *et al.* forthcoming) which has expanded into a larger, wider S&TP corpus, with the addition of a small subcorpus of the spoken BNC[8] compiled by Martin Wynne.

The main corpus is divided into three sections representing different types of modern British English narrative texts, each of approximately 80,000 words. The sections are fiction (subdivided into serious and popular fiction), newspaper reports (subdivided into broadsheet and tabloid newspapers) and biography (subdivided between serious and popular biography and autobiography). There is also the parallel spoken corpus of 80,000 words of modern British spoken English, subdivided into formal and informal contexts. All of the texts have been annotated manually with the tags for different categories of speech and thought presentation listed in Table 6.7. A tag may also take one of the following affixes, providing

Table 6.7 Categories of Speech and Thought Presentation and Corresponding Tags

Category	Acronym
Narrative	N
Narrative Report of Speech	NRS
Narrative Report of Writing	NRW
Narrative Report of Thought	NRT
Narrative Report of Internal State	NI
Narrative Report of Voice	NV
Narrative Report of Speech Act	NRSA
Narrative Report of Writing Act	NRWA
Narrative Report of Thought Act	NRTA
Narrative Report of Speech Act with Topic	NRSAP
Narrative Report of Writing Act with Topic	NRWAP
Narrative Report of Thought Act with Topic	NRTAP
Indirect Speech	IS
Indirect Writing	IW
Indirect Thought	IT
Free Indirect Speech	FIS
Free Indirect Writing	FIW
Free Indirect Thought	FIT
Direct Speech	DS
Direct Writing	DW
Direct Thought	DT
Free Direct Speech	FDS
Free Direct Writing	FDW
Free Direct Thought	FDT

supplementary detail about the context: e (= 'embedded'), Q (= 'with quote'), and h (= 'hypothetical'). See also Leech and Short (1981: 318–51) and Semino *et al.* (forthcoming) for a fuller description of the categories. We have assumed in advance that these categories made up a continuum that covers all types of S&TP.

6.3.1 Problems in the categorization of style in speech and thought presentation

Three major areas have proved to be problematic in the annotation of speech and thought presentation.

Defining the annotation scheme

Stylistic categories have usually not been rigorously defined at a theoretical

level, and there are no precedents for tagsets. It is usually necessary to start from scratch, in developing both a rigorous formal structure of categories and also a tagging formalism. However, the lack of 'baggage' that this implies may be an advantage in some respects – researchers may be able to work relatively free of long-running controversies and ingrained habits.

In the case of the S&TP corpus we have used a tagset derived from what seemed like a relatively clear-cut set of categories put forward in Leech and Short (1981), though it has proved useful to extend and refine it. Some new categories have been added and various aspects of the application of existing ones have had to be rethought. One new category is NRSAP – narrative report of speech act with topic – i.e. where the topic of the speech act is reported, although the form and content of the actual speech act itself is not elaborated. Example 6.4 is a typical example from the broadsheet newspaper section of the corpus (followed by an example of indirect speech for comparison).[9]

In this example, in the sentence *He also called for* ***an immediate end to the fighting***, the reported speech (highlighted) is a nominal complement of the reporting speech. When there is this level of syntactic fusion between the reported and reporting speech it cannot reasonably be considered a case of indirect speech. It is also considered to be different from NRSA – narrative report of speech act – a category reserved for cases where the occurrence of a speech act is reported with no report of the content of the utterance. NRSAP therefore occupies an intermediate position between NRSA and IS. Interestingly it can be seen that in this sector of the S&TP continuum, the criteria separating one category from its neighbours are of different orders. NRSAP is differentiated on the one side from NRSA on the grounds of the amount of the content of the utterance that is

Example 6.4

<sptag cat=NRSAP next=NRS s=1 w=10>
He also called for **an immediate end to the fighting.**
<P>
<sptag cat=NRS next=IS s=0.48 w=15>
Foreign Secretary Douglas Hurd – who flew to Belgrade in a new push for peace – said
<sptag cat=IS next=NRS s=0.52 w=16>
the West was just weeks away from pulling out if the Bosnian Serb warlords rejected peace.<P>
<sptag cat=NRS next=IS s=0.07 w=2>
He warned
<sptag cat=IS next=NI s=0.93 w=28>
that if the warring factions refused to talk, the allies would have no choice but to pull their troops out and lift the arms embargo on Bosnia's Moslems.

reported – i.e. a pragmatic criterion – and on the other side NRSAP is differentiated from IS on mainly syntactic grounds – the level of syntactic fusion between reporting and reported clauses (see Li 1986 on this topic of fusion). In fact, this new category of NRSAP accounted for some 15 per cent of all speech reports in the corpus.

Deciding what segments are to be tagged

In syntactic tagging, the types of units that are being tagged are predefined and central to the theoretical model and to the job of annotation. With stylistic tagging, there is not such an easy equation of surface structure and abstract unit of analysis. Generally we are tagging clauses, but there are problems of various kinds. These include:

(a) *Interpolations* (often one word)

Example 6.5

```
<sptag cat=NRT>
she thinks
<sptag cat=IT>
she wins men's affections
<note desc="interpolated 'like'">
like by sleeping with them all
```

(b) *One word NVs*

The category 'narrative report of voice' (NV) is used to tag instances where the existence of a speech act is reported, but no indication is given of the speech act involved. In Example 6.6, it is not clear whether simply the word *talks* or the clause *After talks in Belgrade* should be categorized as NV:

Example 6.6

```
<sptag cat=NV>
After talks in Belgrade,
<sptag cat=NRS>
Mr Milosevic said
<sptag cat=IS>
he fully agreed with the international peace plan
```

(c) *Embedded quotes*

One mixed form of speech presentation, which is particularly prevalent in broadsheet newspapers, is the use of a few directly-quoted words or phrases in a passage otherwise reported from the viewpoint of the narrator. In these cases a Q is added to the tag to indicate the embedded quote.

While there is a mechanism for representing this it has the disadvantage of not explicitly and separately demarcating the quoted passage from the rest of the unit.

Dealing with overlap and ambiguity

Different amounts of mixed points of view are permissible in different categories. For example, quite a lot of narrative material is allowed to penetrate into categories like NRSA and NRSAP because, with these categories, there is by definition a high degree of authorial intervention. The analyst tagging the text therefore needs to be able not only to interpret the different deictic, lexical and other indicators of linguistic viewpoint, but also to judge the extent of interpenetration of narrative points of view which are permissible in each category.

A formalism for tagging needs to be developed that is powerful enough yet not too complex to prevent manual tagging. It is necessary to account for ambiguous, overlapping, embedded and cleft constructions and other complexities. For feature markup, it has proved preferable to use the Standard Generalized Markup Language (SGML), with guidelines conformant to the Text Encoding Initiative guidelines for feature structure markup, as in Ide and Véronis (1995).

In S&TP tagging, ambiguity is a more common feature than in part-of-speech or syntactic tagging. Utterances are often, and sometimes deliberately, stylistically ambivalent. While there are mechanisms for dealing with ambiguous tags (cf. Section 9.3), it is perhaps necessary here to differentiate different types of ambiguity. There are (at least):

(a) *Genuine stylistic ambiguity* In Example 6.7 (from the spoken corpus, hence no quotation marks) there are not the usual clues of orthography, deixis and tense to tell whether the reported clause is direct or indirect:

Example 6.7

```
he went back to the doctors today
<sptag cat=NRS>
and she said
<sptag cat=IS-DS>
it could take up to six to eight weeks to get out your system.
```

(b) *Fuzzy boundary phenomena between categories* In the Example 6.8 the reported clause is not subordinate to the reporting clause as is normal for indirect speech. Syntactically it is a structure typical for direct

speech, but pragmatically the reader will interpret this as indirect speech. Such a structure, intermediate between DS and IS, would normally be called FIS, except that here we have an explicit reporting clause, making this rather less 'free' than examples that would normally be called FIS. This could be called the paratactic projection of indirect speech (see Halliday 1994 on projection) and it is not clear how to annotate it using our set of categories.

Example 6.8

<sptag cat=IS-FIS>
Taxing the Queen had happened under the Tories,
<sptag cat=NRS>
said Mr Blair's office.

6.3.2 Is automatic annotation possible?

Automatic tagging is difficult because surface syntax is often not a sufficient indicator of stylistic features.

Direct speech may be relatively easy to tag automatically. However, as we have seen above, the disambiguation of indirect speech (IS) from narrative report of speech act with topic (NRSAP) depends principally on the syntactic distinction of whether the reported clause is nominalized or not. So for an automatic procedure to capture this distinction, parsing of the sentence is necessary. The necessity of a prior parse of the text clearly increases the complexity of the procedure.

Another problem, rather more intractable at a linguistic level, is that the indices of reported speech and thought are typically ambiguous. To illustrate this point (Example 6.9), it is only necessary to point out that *verba dicendi* are not always reporting verbs, especially with thought verbs, but also with verbs of saying:

Example 6.9

I **say** today that we have not ever found any any [*sic*] proof of any child abuse out there.

Here what appears to be a reporting clause is really a performative, rather than a report of a prior speech act. Indeed many verbs (e.g. *goes*) can be used to report, but it is often possible to understand them as reports only from an appreciation of the context.

Furthermore, the important and interesting category of **free indirect speech** (FIS) is more or less defined by the absence of formal features to identify it. Although sometimes it may be possible to recognize FIS by the presence of close deictics and interjections (see McHale 1978), there will always be cases, such as the last six lines of Example 6.10 where there are no such surface clues.

Example 6.10

<sptag cat=NRSAT who=K next=FIS whonext=K s=0.7 w=12>
he told a story about one particular commercial that he'd worked on.
<sptag cat=FIS who=K next=DS whonext=M s=5 w=69>
It was a testimonial for a funeral parlour which had dealt with the victims of a forest fire. He needed the sound of a forest fire running under the voice-track, but he couldn't find the effect on file. It was seven at night and the commercial had to be presented at breakfast the next day. In the end he had no choice. He had to create the effect himself.

It seems that given these problems, full automatic annotation would require modelling the entire discourse structure of the text (see Banfield 1973, 1982, Wiebe 1990, Wiebe and Rappaport 1991) and using an intelligent knowledge-based system to attempt to emulate aspects of the real-world knowledge that human speakers routinely bring to bear in order to interpret the stylistic aspects of the message. So tagging is currently carried out manually, and is thus liable to arbitrary subjective variation.[10]

6.4 Conclusion

In this and preceding chapters of the book we have taken a journey which began with the most secure and agreed form of annotation (grammatical word tagging), and ended with probably the least secure type of annotation (stylistic annotation), where the absence of clear and concrete criteria to identify categories inevitably leads to a considerable degree of indeterminacy. These two varieties of annotation are also at the two ends of a scale of automation: although grammatical tagging can already be performed largely automatically, it is not easy to imagine a state of the art where recognition of speech and thought presentation types can be done more than very partially by automatic means. Even here, however, certain well-signalled categories of S&TP, especially direct speech, could be automatically recognized, and greater experience with this work should also

lead to a clearer set of guidelines (analogous to the Annotators' Manual for grammatical annotation) for applying one category label rather than another. We should also be aware, at this stage, of the cumulative effect of annotations coexisting at different levels. It will be seen that there is a particularly basic role for POS tagging and syntactic annotation in the recognition of most of the other categories we have discussed in Chapters 4–6. The next chapter will consider how software undertakes or aids the task of annotation.

Notes

1. The term 'labelling' is often used for the addition of phonetic segmental symbols to a speech corpus, and is convenient to use here for prosodic symbols, as it avoids the issue of whether this is 'transcription' or 'annotation'.
2. The Lancaster/IBM SEC was built as a result of funding, as well as collaboration, by IBM UK Research Centre, Winchester. Later, as a result of a further collaboration between Lancaster and Leeds Universities, funded by the Economic and Social Research Council, the SEC was published as a relational database on CD-ROM (Knowles 1995, Roach and Arnfield 1995).
3. For bibliographical evidence of the range of research based on the LLC so far, see Altenberg (1991, 1995).
4. Stiles has proved the applicability of his scheme to interactions other than doctor–patient interactions by encoding conversations ranging from father/son disputes through to discussions in a science lesson.
5. That is, the elements in the interchange were predictable from a point of view of their speech act function, rather than their form.
6. Of course, as with all corpus annotations, Stiles' annotations are actually imposed interpretations. However, the process is systematic and explicit, and Stiles uses a set of encoding guidelines for speech act annotation analogous to that used to guide syntactic corpus annotation, as described in Section 3.2.
7. This research was supported by a major research grant of the Humanities Research Board of the British Academy (Grant no. M-AN2314/APN3489).
8. On the British National Corpus (BNC), see Chapter 1, n. 2.
9. Note that we are using a mark-up scheme based on SGML (see Section 2.4), as the tags and attributes in Example 6.4 indicate.The relevant sections of the text are highlighted in boldface.
10. See Chapter 17 on consistency of manual annotation.

7

A Hybrid Grammatical Tagger: CLAWS4

ROGER GARSIDE and NICHOLAS SMITH

In this chapter we discuss in detail how a piece of software can carry out automatically one important task in corpus annotation. The task is **part-of-speech (POS) tagging** (also called **word-class tagging**, or **grammatical tagging**); that is, assigning to each word in a text its correct part of speech in context. The result of this task, as a form of corpus annotation, was discussed in some detail in Chapter 2. It usually forms a basis for more sophisticated annotation, such as full syntactic parsing or semantic annotation, and it carries out the useful supplementary tasks of splitting up the text into individual words and sentences.

Most current part-of-speech taggers are **probabilistic** or **stochastic** (see, for example, Marshall 1983, Garside *et al.* 1987, Church 1988, DeRose 1991, Cutting *et al.* 1992, Merialdo 1994); that is, they choose a preferred tag for a word by calculating the most likely tag in the context of the word and its immediate neighbours. At the same time, non-probabilistic or **rule-based** taggers (which began with Greene and Rubin 1971) have been making something of a come-back, with the tagging systems discussed by Brill (1992) and Voutilainen (1995: 165–284). In practice it may be that a **hybrid** system, which combines both probabilistic and rule-based approaches, captures the best of both techniques.

Most serviceable taggers today attain an accuracy in the region of 95–98 per cent. However, what is meant by such a figure is open to variant interpretations. It is probably better to avoid drawing conclusions about the quality of tagging software from comparing crude accuracy rates until we know more about the quality of the linguistic distinctions which the tagger makes, and how consistent analysts have been in checking the accuracy of a tagger (see Chapter 17).[1]

One of the earliest probabilistic taggers was **CLAWS** (Constituent Likelihood Automatic Word-tagging System), developed by UCREL at the University of Lancaster (Marshall 1983, Garside *et al.* 1987). This chapter discusses the current incarnation of this piece of software, CLAWS4; this

could now be considered to be a hybrid tagger, involving both probabilistic and rule-based elements. It has been designed so that it can be easily adapted to different types of text in different input formats.

7.1 Probabilistic and Rule-based Taggers

In natural language processing by computer up to the late 1970s, part-of-speech tagging was seen as a by-product (and not a very interesting one at that) of full syntactic parsing. However the **TAGGIT** program (Greene and Rubin 1971) introduced the idea of providing a text corpus annotated with part-of-speech information as a useful tool for linguistic research. This was the **Brown Corpus** of one million words of written American English, collected in the early 1960s. The tags assigned were from a set of some 77 tags (the Brown tagset). The basic idea in the TAGGIT program was to associate with each word a set of potential tags, and then use the context to choose the correct one. The mechanism for the initial assignment of tags to a word relied on a lexicon, a word-ending list, and a set of other rules for dealing with capitalized words, hyphenated words, etc.; as we will see, this general type of mechanism is used in CLAWS (and indeed in other probabilistic taggers). The contextual disambiguation was carried out in TAGGIT by means of context-frame rules. A context-frame rule was a rule, designed by a linguist based on observation of data, which specified some information about a potential tag in the context of up to three tags on either side – the rule could specify that the potential tag was the correct one in context, or that the potential tag was impossible in this context (so that one of the other potential tags must be the correct one). All the tags being used for contextual clues in a context frame rule had to be unambiguous, so the context frame rules had to be tried several times, in the hope that disambiguating a tag at some point in a sentence would allow a context frame rule now to be applied to disambiguate another tag in the sentence.

After this pioneering tagger using the rule-based paradigm, interest passed in the early 1980s to probabilistic taggers. The general idea is that, if we have a sequence of words, each with one or more potential tags, then we can choose the most likely sequence of tags by calculating the probability of all possible sequences of tags, and then choosing the sequence with the highest probability. Thus, if we have a sequence of words w_1, w_2, ... , w_n, the goal of tagging is to select the most likely sequence of tags t_1, t_2, ... , t_n associated with those words, and we assume that this is the correct sequence. The statistical model which has been most used in POS tagging is that of the **hidden Markov model** (HMM) (see Poritz 1988). We can

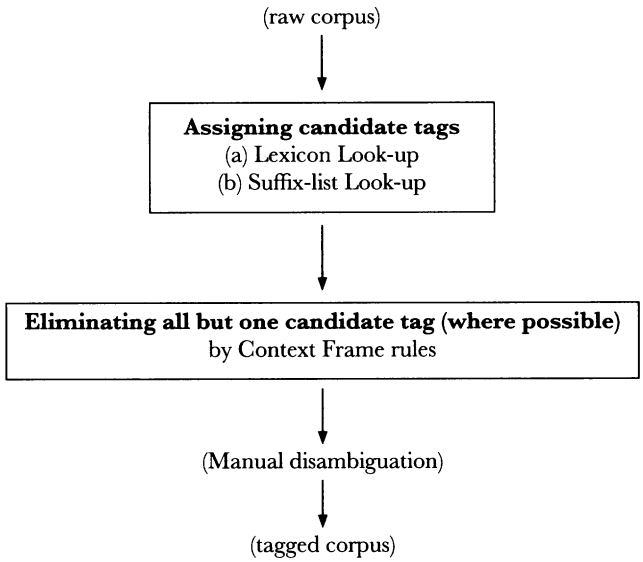


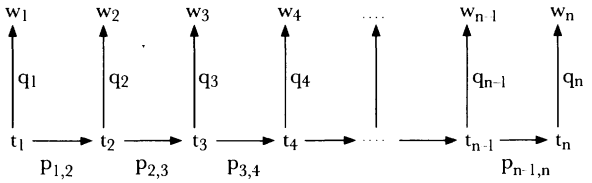
Figure 7.1 Diagram of the TAGGIT system processing the Brown Corpus

directly observe the sequence of words, but we can only estimate the sequence of tags, which is 'hidden' from the observer of the text; hence the term 'hidden Markov model' is appropriate. A HMM enables us to estimate the most likely sequence of tags, making use of observed frequencies of words and tags (in a **training corpus**).

The probability of a tag sequence is generally a function of:

- the probability that one tag follows another; for example, after a determiner tag an adjective tag or a noun tag is quite likely, but a verb base form tag is less likely. So in a sentence beginning *the run...*, the word *run* is more likely to be a noun than a verb base form.
- The probability of a word being assigned a particular tag from the list of all possible tags for the word; for example, the word *over* could be a common noun in certain restricted contexts (of cricket reports), but generally a preposition tag would be overwhelmingly the more likely one.

On page 105 we have the sequence of words $w_1, w_2, \ldots, w_n$ and a possible sequence of tags $t_1, t_2, \ldots, t_n$. The probability of this sequence of tags makes use of estimates of the tag transition probabilities shown as $p_{1,2}$, $p_{2,3}$, etc. in the diagram below. In the simplest case these estimates are derived from frequencies of tag pairs ('bigrams' in the training corpus). The second type of probability that enters into an HMM is that labelled q_1, q_2, etc.



in the diagram. This is the probability that the word w_i will be associated with the tag t_i. Obviously, with a sufficiently large training corpus, it will be possible to estimate the q_i from the relative frequency with which any particular word and tag associate with each other.

This model is a *first-order* HMM, since the estimates used for the tag transition probabilities are derived from bigrams; that is, we have estimated the likelihood of a particular tag occurring given only the preceding tag. A second-order HMM would use tag transition estimates derived from trigrams; that is, we estimate the likelihood of a particular tag occurring given the preceding *two* tags. This is clearly a more refined estimate of the probability of one tag following another, but we have to calculate a much larger set of estimates and need a correspondingly larger training corpus.

It is clear that the HMM model descibed above is no more than a rather rough approximation to the problem we are trying to solve when we tag a corpus. For one thing, it treats the tag sequence as an abstraction from what the words are; it ignores, for example, the problem of how to deal with idiosyncratic word sequences or multiwords like *as well as*. Secondly, and more notably, it ignores any grammatical constraints on the word-class of a word, apart from constraints derived from its immediate neighbours. Nevertheless, in spite of its manifest theoretical limitations, the HMM approach to tagging is surprisingly successful, and various taggers of this general design result in a 95–97 per cent accuracy rate. The Xerox tagger, which is of this general design, is discussed further in Section 10.2.

After Greene and Rubin's TAGGIT tagger, the next rule-based tagger to attract serious interest was that of Brill (1992, 1994), and this illustrates how the rule-based approach contrasts with the probabilistic approach. Like a probabilistic tagger, Brill's tagger requires a training corpus and, using this, the tagger works by 'automatically recognizing and remedying its weaknesses, thereby incrementally improving its performance' (Brill 1992). The first step is to apply the most likely tag for each word (i.e. 'most likely' without reference to any left or right context, what is called the 'unigram' probability) – for this step, and this step alone, quantitative information is needed. The training corpus, tagged in this way, has a relatively high accuracy (in the region of 90 per cent), and the task then

is to improve this result by iteratively applying a set of patching rules of the following form (excerpted from Brill 1992):

change tag *a* to tag *b* when:
1. The preceding (following) word is tagged *z*.
2. The word two before (after) is tagged *z*.
3. One of the two preceding (following) words is tagged *z*.
4. One of the three preceding (following) words is tagged *z*.

These rules, as can be seen, make use of information available in the immediate context of the 'target word', although longer-range rules are also possible. The tagger looks at the application of each rule of this type to the training corpus, and computes the number of errors remedied by its application and the number of new errors introduced. For example, the most successful of Brill's (1992) patches was one which changed the tag 'infinitive marker' to 'preposition' when an article followed (e.g. in *to the*, the word *to* could scarcely be an infinitive marker). The procedure of learning the patches is iterative, and after each run the most successful patch is added to the list of patches. The 'patching rules' are ordered in terms of the net improvement they achieve (i.e. the size of the difference between tags corrected and tags wrongly corrected). If one wishes to tag a new corpus, the patches (after the basic tagging has been done) are applied in order of success-rate; but if a patch changes tag *a* to tag *b*, it applies only if there is some instance in the training corpus of the word in question having the tag *b*.

Brill's rule-based tagger has produced results comparable to HMM taggers, and therefore has challenged the orthodoxy (which had been growing up in the early 1990s) that statistical methods outperform rule-based methods. At the same time, statistical taggers do not seem to be making significant progress towards the goal of 100 per cent success, and this may be because they are lacking in the kinds of grammatical knowledge about language which linguists take for granted. There may be a plateau which probabilistic taggers have reached, and there may be limits to how far one can go without a richer kind of linguistic knowledge.

This may be borne out by the fact that the one notable improvement on the 3–5 per cent error rate claimed up to now has been a grammar-based system, **ENGCG** (the English Constraint Grammar of the Helsinki team of Karlsson and Voutilainen – see Section 3.3.5). Certainly the grammar- or rule-based approach can be taken much further than was previously thought.

In fact many systems are not quite so 'pure' as the above discussion implies, and already there is some combining of strategies. Nearly all probabilistic taggers have sets of heuristic rules or guessers dealing with

unknown words, while some rule-based systems use a limited amount of frequency information, as we have seen with Brill's (1992, 1994) system. A systematic attempt to integrate the two approaches is described in Tapanainen and Voutilainen (1994). Their experiment involved tagging a text with both a rule-based tagger (ENGCG – see Section 3.3.5) and a probabilistic tagger (the Xerox tagger – see Section 10.2) and aligning the outputs from the two programs. Where ENGCG succeeds in fully disambiguating a word its analysis is preferred to that of the Xerox tagger; where ambiguity remains in the former, it is resolved by accepting the disambiguation of the latter.

7.2 The CLAWS Tagger

The CLAWS tagger, discussed in detail in the remainder of this chapter, could be considered to be a hybrid tagger, involving both probabilistic and rule-based elements, even in its earliest form (CLAWS1 – Marshall 1983, Garside *et al.* 1987). The probabilistic element was an approximation to a HMM tagger. In one respect it was less that a HMM tagger; instead of using probabilities of word-tag association (the probabilities in the diagram on p. 105), it relied on human judgement of frequency applied to tags for ambiguous words in the lexicon. A three-point scale was used – common, rare (less than 10 per cent of the word occurrences were expected to receive this tag), very rare (less than 1 per cent of the word occurrences were expected to receive this tag). The reason for adopting this expedient was that the training corpus was judged not to be large enough to provide reliable word-tag association statistics.

In another respect, CLAWS1 was already more than a HMM tagger: it contained an embryonic rule-based component, the so-called 'idiomlist' (see Blackwell 1987) which enabled it to carry out exceptional taggings; for example, to tag **multiwords** (see Section 2.2) such as *as for* or *in order that* as single tokens, or to identify common tag sequence constraints which departed from what could be expected using the HMM mechanism (for example *dining room* tagged as NOUN–NOUN rather than ADJECTIVE–NOUN).

The 'idiomlist' component has been enormously expanded in later versions of CLAWS, so that the term idiomlist (never very satisfactory, as it suggests that it searches only for patterns which are linguistic idioms) should be replaced by the term **rule-based component**. This component in current versions of CLAWS not only identifies exceptional sequences (for example multiwords, foreign expressions and complex names of various kinds), it also carries out a significant role in disambiguation, sometimes preempting, sometimes correcting the probabilistic processing

of major categories such as infinitivals and past participles, which are liable to cause trouble for a tagger relying on probabilistic resources alone.

The first version of the CLAWS tagging system (which was subsequently named CLAWS1) was developed at Lancaster over the period 1980–83. It was developed as part of a project to assign part-of-speech information to the LOB (Lancaster-Oslo/Bergen) Corpus, a one-million word corpus of British English designed to match the Brown Corpus in size, scope and structure. The tagset used, which became known as the CLAWS1 tagset, was a development of the Brown tagset, using about 135 tags.

The second version of CLAWS (CLAWS2) was developed over the period 1983–86. One special feature of the LOB Corpus was that certain features such as sentence breaks were explicitly marked in the text. CLAWS2 was developed as a tagger which could be run over general text, without explicit mark-up of this kind. The tagset used, the CLAWS2 tagset (with 132 tags), was a revision and refinement of the CLAWS1 tagset, based on experience with using this original tagset.

The development of CLAWS4 began in 1988,[2] and a number of versions of this software have been produced; the latest version (late 1996) being version 17. One development has been the separation of the CLAWS software from the tagset used. As mentioned above, earlier versions of CLAWS were closely designed round a specific tagset. However CLAWS4 was developed to tag the one hundred million word British National Corpus (BNC), and for this two tagsets were to be used:

- a detailed tagset (C7) of 146 tags for a two million word **sampler** corpus, and
- a less refined tagset (C5) of 61 tags for the rest of the corpus.

The opportunity was taken to decouple the program code from the tagset, which is now read in as part of the resources required for a particular tagging task.

7.3 Input Issues

The first version of CLAWS was designed specifically for the LOB Corpus, with its special notation for representing sentence breaks, changes of typeface, special characters, etc. (see Johansson *et al.* 1978). CLAWS2 was designed to cope with text which used normal orthographic conventions, but it soon became clear that CLAWS would have to be able to cope with representation of special characters such as accented letters, and of the structure of a text – for instance, it is fairly common for certain parts of

a document not to be text for tagging, and these have to be marked so that they can be ignored.

When CLAWS4 came to be redesigned so that it could be used for the British National Corpus project, it was decided to move over to using SGML (Standard Generalized Mark-up Language: see Section 2.4) to represent all features of a text. This is the normal default assumption in current use of the CLAWS system, although it is possible also to process files in plain ASCII. In a number of recent UCREL projects, a pre-processing program running before CLAWS translates a text from a different format into the standard set of SGML tags and entities used by CLAWS. For example in one project there are a number of special purpose programs to take files of text from a variety of sources (including the World Wide Web) and with a number of different formats to represent emphasis, quotation marks, special characters, etc., and to convert them into the standard format required by CLAWS.

One of the resources read by CLAWS at the beginning of a run is a list of all valid SGML tags and entities together with the action to be taken on recognizing them. It is of course possible to supplement the standard set of tags and/or entities for processing a particular text. If CLAWS encounters a tag or entity not in these tables, it displays an error message. The default action for a valid SGML tag is for CLAWS simply to ignore it, copying it from the input to the output with an associated **null** part-of-speech mark. This would be the normal case, for example, for tags marking text structural divisions (chapters, paragraphs, etc.) or denoting typeshifts (bold, italic, etc.). Alternative actions which can be specified are:

- to ignore all text bracketed by a particular pair of tags – this was not used in the BNC data, but might apply, for example, to editorial notes inserted in a text.
- to treat a particular tag as the start of the taggable text. In the BNC written text was always enclosed within <text> ... </text> markers, and spoken text was enclosed within <stext> ... </stext> markers. If CLAWS reads an SGML-conformant text it always reads up to the first such start tag before processing any text, stopping when it meets the corresponding end tag, and starting to process text again if it meets a further start tag.

One problem with skipping everything up to the first start tag, is what to do with the characters passed over, since we presumably wish to end up with a single text file containing all the original information plus the additional part-of-speech information. Since the first version of CLAWS its output format has been rather restricted; what is referred to as 'vertical output' (see Section 7.6), with a single orthographic unit on each line, and a fixed format of text line reference, word (up to 25 characters), subsidiary

information (enclitic markers, error markers, CLAWS decision codes, etc.), and the part-of-speech tags. A number of programs have been written at UCREL (including several editors) which expect text in this format, so it was difficult to change it to a more flexible format. Any data that do not fit within this rigid format, including the SGML **header information** which precedes the text proper in an SGML document (often several thousand characters long in the BNC) are therefore copied to a supplementary free-format output file, and a marker is placed in the normal CLAWS output file indicating their offset and length in the supplementary file. Thus the two files could be merged back together without loss of information, after the post-editing and other post-processing had taken place.

This supplementary file also solved a problem never satisfactorily resolved in earlier versions of CLAWS; what to do about long words. In earlier versions of CLAWS a long word (that is, one longer than 25 characters) was simply truncated, with an error message. Now it is simply inserted in the supplementary file and a suitable pointer inserted in the normal CLAWS output file.

The table of SGML entities (marked by an opening '&' and a closing ';') indicates for each valid entity what class of character it represents, and CLAWS takes an appropriate action for each class. Some of the more commonly encountered classes are:

- accented letters such as É (representing an upper case letter) or ĉ (representing a lower case ĉ). Since words containing accented letters may be naturalized into English with or without accents, a word containing entities of this class is looked up in the lexicon with and then without the accents. The SGML entity table entry for this class specifies what unaccented character or characters correspond to this accented letter – thus É corresponds to 'E', and œ corresponds to 'oe'.
- certain classes of character can be specified as of a type of character to be ignored. Thus, the word Unix© is treated as if it were 'Unix', since the character © (a copyright sign '©') is treated as a character to be ignored.
- a further class of SGML entities, including such entities as ⅓ for the fraction one third, is specified as to be treated as part of a numerical value.

An example of an SGML input text to CLAWS4, illustrating some of the features mentioned above is

```
<text>
The na&iuml;ve cat sat on the <hi rend="italic">Persian</hi> mat.
</text>
```

which represents the sentence 'The naïve cat sat on the *Persian* mat.'

7.4 Tagging Individual Words

In this section we discuss how a set of one or more potential part-of-speech tags is associated with each individual orthographic unit – a word or other sequence of graphic characters considered as a unit.

The assignment of tags is treated as a sequence of tests of the current orthographic unit. If a test succeeds then an appropriate set of tags is assigned; if not, the next test is applied. The sequence of tests is as follows:

1. First a number of tests are carried out for orthographic units of certain special types:
   (a) for long words (i.e. words over 25 characters long), which are treated by default as common nouns
   (b) for truncated words, which are given the unknown tag (FU in C7 or UNC in C5). In the spoken part of the British National Corpus truncated words (e.g. *never* truncated to *nev* at a point where an utterance is interrupted) are bracketed by the SGML tags <trunc> ... </trunc>. In tagging other spoken corpora, such as the **COLT corpus** of London teenage discourse[3] where truncated words are marked by a trailing =-symbol, a pre-processing program is used to map the notation into the BNC format
   (c) and for clitics, such as the *'ll* of *he'll* (see Section 2.2)
2. Next the full word is first looked up in the main CLAWS **lexicon**. The look-up procedure is simplified by converting the word to be looked up into a standard form (all lower case, no abbreviatory full stops). If an entry is found in the lexicon for this word, then it contains a list of potential tags for the word, but the tags are annotated with the type of orthography to be expected if the word is to be allocated this tag. A filter process uses the orthography of the word in the text to retain only the appropriate subset of the tags. Thus a lower-case word would retain only those tags marked as appropriate for lower case, while a word with an initial capital would retain the tags marked as expecting an initial capital (types of proper noun, for instance). But those tags marked as appropriate for lower case are also retained, since it is common to find words of this type capitalized at the beginning of sentences or in headlines. It would have been possible to apply this filtering process more selectively, since, for example, in the BNC, headlines are marked with a special SGML tag. However, the capitalization process

is more widespread than this; furthermore the SGML **headline** tag has not always been found to be present where expected

3. Tests are carried out for dropped initial *h* and final *g*, succeeding only if the resulting word is found in the lexicon (for example *'ouse* and *anythin'*). This tends not to be very useful for the transcription of the spoken part of the BNC, but it is quite useful in representation of spoken dialogue in written texts.
4. Tests are then carried out for words with a trailing *s*; this is stripped off, and spelling rules are used to obtain a suitable base form, which is then checked against the lexicon. A filtering process retains only those tags consistent with a trailing *s* (plural nouns of various types, and third person singular forms of verbs).
5. Next there are a number of tests for special orthographic units of various sorts – this set of tests has tended to be expanded fairly frequently as new classes of orthographic unit are recognized by the analysts post-editing the output text. This step deals with
   (a) individual letters, numbers of various types, Roman numerals
   (b) words of the form *A/B*. The two portions are looked up in the lexicon, and the common set of tags from the two parts – words *A* and *B* (if any) – are assigned to the word *A/B*
   (c) formulae, recognized by containing a mixture of letters and numbers, or containing special characters like +, are assigned a **formula** tag (**FO** in C7).
6. Words containing a **hyphen** are dealt with at this stage (if the word has not already been dealt with, by appearing in the lexicon). There are three main procedures:
   (a) since certain words can appear with or without hyphens, and since extra hyphens can sometimes be inserted because of line-breaks in the text, the hyphens are first removed and the word looked up again in the lexicon
   (b) a second procedure recognizes certain prefixes which can be added to a word without changing its grammatical class. Thus a prefix of this type can be recognized in *counter-attack*, and the appropriate tags extracted from the lexicon entry for *attack*
   (c) finally, a hyphenated word *A-B* is broken into the two parts *A* and *B*, these parts looked up in the lexicon, and an attempt made to construct tags for *A-B* from the tags for *A* and *B*. Thus **past tense of verb** followed by **adverb or preposition** can result in **adjective** (as in *fed-up*).
7. The next step is an attempt to predict the appropriate tags by considering the ending of the word:
   (a) First the word-ending *er* is treated specially. The ending is stripped

from the word, and the result looked up in the lexicon. Essentially verb tags associated with the stem indicate a common (agentive) noun for the word (e.g. *listener*), and adjective tags indicate a comparative adjective (e.g. *odder*)

(b) A list of **word-endings** with associated tags is then searched for the word, and the longest match found. As with searches in the lexicon a filtering process is again used, so that a word with a particular ending could have distinctive tags if it appears with or without an initial capital. (For example, *-man* with a word-initial capital is likely to be a proper noun, but not if the word begins with a small letter – compare *Bowman* and *bowman*)

(c) Finally a trailing *s* is stripped, and the resulting stem looked up in the list of word endings.

8. The final step is a default procedure, if all the above tests have failed. Any orthographic unit reaching this point is allocated the default set of potential tags – noun, verb, adjective, adverb.

7.5 Using Probabilities

The result of the procedures described in the previous section is that each word in the text receives one or more part-of-speech tags. The task of the probabilistic part of CLAWS is to choose a single preferred tag in cases where a word has more than one. In these cases CLAWS in fact ranks all the potential tags, from most likely to least likely, assigning to each a probability of the tag being the correct one. This probability figure can be used to estimate the likelihood of the preferred tag being the correct one, and allows the introduction of **portmanteau tags** (see Section 9.3).

As mentioned in Section 7.1, the basic mechanism used by CLAWS is to estimate the likelihood of tags over a sequence of words starting with a word with a single unambiguous tag, continuing over a sequence of one or more words with more than one potential tag, and finishing again with a word with a single tag. Since punctuation marks are unambiguously tagged in the CLAWS system, in the worst case the sequence of words would be a complete sentence, but it is usually shorter. In principle CLAWS then considers each sequence of possible tags for this sequence of words, estimates the probability of that sequence, and then chooses the sequence with the highest probability. The probability of a sequence is calculated from:

- the conditional probability $P(t \mid t_{-1}\, t_{-2} \ldots)$ of a particular tag t given that the preceding tags were t_{-1}, t_{-2}, etc., and

- the conditional probability P(w | t) of a particular word w, given that the associated tag was t.

Consider an example, where words w_0 and w_4 are unambiguously tagged t_0 and t_4 respectively, and the intervening words w_1 to w_3 have two or more potential tags each:

w_0	w_1	w_2	w_3	w_4
t_0	t_{11}	t_{21}	t_{31}	t_4
	t_{12}	t_{22}	t_{32}	
			t_{33}	

Then the probability of the words w_0 to w_4 being tagged t_0, t_{11}, t_{21}, t_{31}, t_4 is the expression

$$\begin{aligned} & & & & P(w_0 \mid t_0) \times \\ P(t_{11} \mid t_0 \ldots) & \times & P(w_1 \mid t_{11}) & \times \\ P(t_{21} \mid t_{11} \ldots) & \times & P(w_2 \mid t_{21}) & \times \\ P(t_{31} \mid t_{21} \ldots) & \times & P(w_3 \mid t_{31}) & \times \\ P(t_4 \mid t_{31} \ldots) & \times & P(w_4 \mid t_4) & \end{aligned}$$

and a similar expression can be calculated for each possible tag sequence. Establishing the most probable sequence in this way can result in a large amount of calculation, especially for a long sequence of ambiguous words each with several alternative tags. However, there is a procedure, called the **Viterbi alignment**, which can reduce sharply the amount of effort required:

1. We can calculate the probability of the most likely (indeed the only) path from t_0 to each of t_{11} and t_{12}. For example, the probability of the former is $P(w_0 \mid t_0) \times P(t_{11} \mid t_0 \ldots) \times P(w_1 \mid t_{11})$.
2. We can then calculate the probability of the most likely path from t_0 to each of t_{21} and t_{22}:
   (a) consider the path from t_0 to t_{21}; it goes through either t_{11} or t_{12}. The probability of the path going through t_{11} is the probability of the most likely (the only) path from t_0 to t_{11} × $P(t_{21} \mid t_{11} \ldots) \times P(w_2 \mid t_{21})$
   (b) we can similarly calculate the probability of the path through t_{12}, and then choose the path to t_{21} with the highest probability. We record this highest probability, together with information as to whether the path went through t_{11} or t_{12}
   (c) we use the same mechanism for choosing the most probable path to t_{22}.
3. We can then calculate the probability of the most likely path from t_0 to each of t_{31}, t_{32} and t_{33}. This calculation looks back to the information stored for the possible paths leading up to word w_2, but no further.

4. We can continue with this **forward** calculation, until we reach the end of the ambiguity. At the end we know the probability of the most likely route, and which was the best choice of the last ambiguous tag (here t_{31}, t_{32} or t_{33}). But the information stored with this tag enables us to find the best choice of the next to last ambiguous tag (here t_{21} or t_{22}).
5. We can make a **backward** pass, extracting the best choice of tag for each word as we go.

The above Viterbi calculation tells us the most likely tag sequence, and what its probability is. We may also want the probabilities of the individual tags. For example, the most likely path in the above example might be t_0, t_{11}, t_{21}, t_{31}, t_4 with a probability of 0.4; but there might be two paths through tag t_{12} each with a probability of 0.3 (and all other paths have negligible probability). Then the individual probability for the tags t_{11} and t_{12} is 0.4 and 0.6. We can calculate this by an extension of the above Viterbi alignment; on the backward pass we make a similar calculation to that on the forward pass, and from this we can calculate the individual probabilities (see Jelinek 1976, 1990).

CLAWS carries out the above calculations for all sequences of one or more ambiguously tagged words, and reorders the tags by decreasing individual probability, but with the tag on the most likely tag sequence first; the first tag in the list is CLAWS' preferred tag for this word. It is possible (as indicated in the example above) for the tag with the highest individual probability not to be the tag on the most likely sequence, but it is rare.

The probability calculation makes use of information about the likelihood of one tag following another. CLAWS4 is set up to allow the likelihood figures to make use of only the preceding tag, $P(t \mid t_{-1})$, or of the two preceding tags, $P(t \mid t_{-1}\, t_{-2})$. Most of the recent work with CLAWS4 has made use of only the bigram statistics $P(t \mid t_{-1})$. As part of the production of the British National Corpus a two-million word **sampler corpus** was constructed, and this was manually post-edited so that only a very small percentage of errors are likely to remain in its tagging. This has been used as **training data**, to generate a set of bigram probabilities of one tag following another. In fact the sampler corpus is made up of one million words of written text and one million words of spoken text, so two separate probability matrices have been generated, one for tagging written material and one for spoken material.

There is a problem with the calculation of probabilities of tag sequences. If a certain tag transition has never been seen in the training data, then any tag sequence containing this transition will have a probability of zero, and will never be considered. A probabilistic tagger works on the principle that all tag sequences are possible, but some are more

probable than others. The CLAWS system is therefore set up so that any tag transition which does not occur in the training data is given a very small probability, so the transition is not treated as completely impossible.

The calculation of the best tag sequence also makes use of the probabilities P(w | t), that a particular word is associated with a particular tag. CLAWS in fact stores information about a particular tag being associated with a particular word P(t | w), and uses Bayes' theorem (see Jelinek 1990) to calculate P(w | t). There are two mechanisms in CLAWS4 for supplying these probabilities:

1. As with the bigram information, figures for word-tag associations can be extracted from a corpus of correctly tagged text. Current versions of CLAWS make use of a lexicon induced from the BNC written and spoken sampler corpora, and this has word-tag association figures for all words which appear sufficiently often in the tagged text (see Section 9.2.3).
2. The word-tag association information is likely not to be useful for words which occur only infrequently. Further, it is difficult to arrive at suitable frequency figures for words which have been assigned a set of potential tags as part of some rule-driven procedure, for example for dealing with hyphenated or capitalized words. For this reason CLAWS4 has a second mechanism for indicating crude frequency estimates in cases where good frequencies are not available; in earlier versions of CLAWS *all* word-tag association information was of this type. If good frequency information is unavailable, a linguist can indicate in the lexicon that for a particular word a particular tag is unlikely (nominally less than 10 per cent chance, indicated with a '@'-character) or very unlikely (nominally less than 1 per cent chance, indicated with a '%'-character). Similarly some of the rule-driven procedures deliver frequency estimates of this crude form.

The CLAWS4 probabilities are all obtained by extraction from text corpora which have been corrected by hand. There is another mechanism by which probabilities can be estimated. If we start off with a set of tag transition and word-tag probabilities, and with a corpus of text (without part-of-speech annotation) it is possible to perform an iterative procedure called the **forward-backward algorithm** which adjusts the probabilities a bit at a time in the light of possible tag sequences estimated to occur in this training data. Thus an initial set of estimates of probabilities is adjusted in the light of a quantity of training data of an appropriate type to give a more accurate set of probabilities which can be used on other texts of the same type (see Jelinek 1990 and Chapter 10 for more details of this **self-organizing methodology**).

7.6 Using Contextual Patterns

Early in the development of the CLAWS1 system, two problems were noticed with the mechanism described above:

1. Some text items are traditionally written as two or more separate orthographic words, but function as a single grammatical unit; obvious examples are multiword prepositions such as *according to* (see multi-words, Section 2.2). The UCREL team came to refer to the tags associated with these multiword units as **ditto-tags**, since a sequence of orthographic words would receive the same tag.
2. There were some segmented patterns of words which the probabilistic mechanisms described above did not handle very well, and which could be handled by searching for a few simple patterns of words and/or tags.

It was therefore decided to write a simple pattern matching module, which would run immediately before the Viterbi alignment procedure. There would be a small number of patterns (in the first version of CLAWS, some 150 patterns) each with an associated action – to insert one or more tags on one or more words matched by the pattern. Although the ditto-tag problem could have been solved by extending the lexicon to include multiword units, it was decided to use the contextual pattern matching module for this task as well.

The contextual pattern matching mechanism has been extensively developed in more recent versions of CLAWS. A pattern to be matched consists of a sequence of two or more elements, to be matched to a sequence of two or more words in the text. An element to be matched can consist of any of the following:

- a word (for example *according*), a regular expression representing a word (for example any word ending with *-ing*), a particular word with initial capitalization (thus a pattern element *Times* would not match *times*), any word with an initial capital (this is useful for matching the open-ended portion of certain types of geographical naming expressions, for instance), or any of a list of similar words (this allows multiple patterns to be encoded more concisely)
- a part-of-speech tag (for example any word assigned a potential adjective tag), a regular expression representing a tag (for example any tag starting with an N, indicating a noun tag of some form), or any of a list of possible tags
- an indication that the match must *fail* – for example, it is possible to search for a pattern of words or tags *not* preceded by some part of the verb *to be*

- an indication that a particular pattern element is optional (a common element useful in correcting verb patterns allows an optional intervening adverb or the word *not* or *n't*)
- an indication that an optional element can be repeated a number of times (it is, for example, possible to indicate that up to three words can occur between the two parts of a pattern).

A rather rebarbative syntax has grown up over the years of CLAWS development for indicating all the above types of pattern element. More recent developments, particularly the Template Tagger described in the next chapter, have cleaned up and extended the syntax to allow more powerful matches than are currently possible with CLAWS. An example would be the possibility of defining a named set of words which could be quoted in a number of different rules; in CLAWS the list of words would have to be written out in each pattern which required them.

There is a problem in the CLAWS contextual pattern matching (which re-occurs in other pattern matching programs; see Sections 8.3.3 and 9.2.5) to do with dealing with the overlap between patterns, or the decision of which is the preferred pattern if several match simultaneously (and given that the application of the actions of one matching pattern might cause other patterns to cease to match). CLAWS has a conceptually simple mechanism for dealing with multiple matches:

- the text is scanned from beginning to end, and each pattern is tried at each word position in the text with all possible structures
- if there are more than one matching patterns starting at a particular point, then a score is calculated for each such pattern and the actions of only the highest scoring pattern are carried out. The score is based on the type of match (for example, an element matching on a word scores higher than a match on a tag, and a pattern most of whose matches are on words scores higher than one most of whose matches are on tags), and then on length of match (a longer pattern scores higher than a shorter one). Thus *as well as* beats *as* **Adjective** *as* (by the first criterion) and beats *as well* (by the second criterion)
- when a pattern is chosen by the above criterion, then all other patterns commencing at the same point are abandoned. Furthermore, all patterns which begin within the scope of the matching pattern are abandoned, and the pattern matching recommences immediately beyond the matched pattern. Thus it is not possible with this mechanism to recognize a pattern within the scope of another pattern; for example, a multiword adverb (recognized with a ditto-tag pattern) within a verbal pattern requiring an adverb.

To provide more flexibility the pattern matching in the latest versions of CLAWS is divided into a number of passes. There are two passes before the Viterbi probabilistic disambiguation described in the previous section and two afterwards. The idea is that most multiword units requiring a ditto-tag will receive one in the first pass, and then the results of this pass are available to the second pass. While patterns in the first two passes match *any* potential tag, those in the final two passes match only on the tag preferred by the Viterbi process.

7.7 Conclusions

The result of running CLAWS4 over the text displayed at the end of Section 7.3 is as follows:

```
**6;0;START   NULL
---------------------------
The           AT0
na&iuml;ve    AJ0
cat           NN1
sat           [VVD/91] VVN@/9
on            [PRP/90] AVP@/10
the           AT0
**18;7;hi     NULL
Persian       AJ0
</hi>         NULL
mat           [NN1/99] AJ0@/1 VVB@/0
.             .
**8;26;text   NULL
```

The first column represents the words of the text; items commencing ** indicate a reference to a particular position in the supplementary file (here corresponding to most of the SGML tags). The second column represents the part-of-speech tags assigned by CLAWS, with the preferred tag at the left. Where there is a choice the numbers indicate the percentage likelihood of the individual tag, and the square brackets indicate the preferred tag sequence. SGML tags are given the special part-of-speech mark NULL. The dashed line indicates the insertion by CLAWS of a sentence break. The actual output from CLAWS also includes line reference numbers, tagging decision codes and other subsidiary information.

Currently, CLAWS4 operates with an accuracy rate of some 96–97 per cent across the whole range of texts in the BNC. If manual post-editing is required, an X-Windows-based editor Xanthippe (see Section 13.3.1) provides a user interface onto the (vertical format) text, allowing a correct tag

to be promoted to the preferred tag position or a new tag inserted from a panel of options. Other facilities allow the editor with a few key-strokes to insert or delete sentence breaks, split or join words, or modify the ditto-tagging.

A final program in the suite reformats the output from CLAWS, whether post-edited or not, into normal **horizontal running text** with the part-of-speech tags added; for the BNC the tags are represented as SGML entities. At this stage the characters from the supplementary file are inserted into the output, resulting in a text such as:

```
<text>
<s>
<w AT0>The<w AJ0>na&iuml;ve<w NN1>cat<w VVD>sat<w PRP>on
<w AT0>the<hi rend="italic"><w AJ0>Persian</hi><w NN1>mat<c PUN>.
<s>
</text>
```

In the introduction it was stated that most of the BNC was tagged with a C5 tagset of some 61 tags. In fact it was tagged with a slightly larger tagset: the additions were **process tags**, making distinctions which were useful at the disambiguation stage, but not required in the final result – the mapping to remove these extra tags is performed at the final post-processing stage, as is the introduction of **portmanteau tags**; that is where the CLAWS system is unable reliably to decide between two tags, and consequently both tags are assigned to the word output. This final program decides whether a portmanteau tag is appropriate based on the individual word probabilities calculated by the Viterbi processing described in Section 7.5; the issue of portmanteau tags is discussed further in Section 9.3.

This chapter has described the general structure of the current version of the CLAWS tagging system (CLAWS4), incorporating both a basic probabilistic Viterbi process and a supplementary rule-based set of components, capable of assigning part-of-speech marks to general text with an overall accuracy rate in the region of 96–97 per cent. The next chapter describes the Template Tagger, a program which extends the pattern matching techniques of Section 7.6, to insert further grammatical annotation or to correct annotation (such as part-of-speech information) which has already been inserted. Chapter 9 describes in more detail how the CLAWS program was adapted for use in tagging the British National Corpus with its wealth of different text types.

Notes

1. It should be noted, however, that Voutilainen uses a more complex measure of tagging success, derived from information retrieval, calculating two

percentage figures known as **precision** and **recall**. Recall is the extent to which all legitimate readings are found in the output of the tagger – allowing, that is, for ambiguous taggings of one word. Precision is the extent to which illegitimate readings are discarded from the output (see Voutilainen 1995: 172). Optimally, both measures should be 100 per cent. Voutilainen (1995: 275) records the following impressive result of one experiment: Recall 99.77 per cent; Precision 95.54 per cent.

2. The name CLAWS3 was applied to a modified version of CLAWS2 which involved attempts at verb subcategorization. It never became a fully developed system.
3. The COLT corpus was collected and transcribed at Bergen (see Haslerud and Stenström 1995). Part of it was incorporated into the BNC as part of the spoken material, but an enhanced version, in a more detailed transcription, is being grammatically tagged at Lancaster, using CLAWS4.

8

How to Generalize the Task of Annotation

STEVE FLIGELSTONE, MIKE PACEY
and PAUL RAYSON

8.1 Introduction

In the last 20 years, UCREL's principal technique for automatic grammatical analysis has been a probabilistic one. Training methods for this technique usually rely upon large bodies of text analysed in advance either completely by hand or by machine and then corrected by grammarians (see Garside *et al.* 1987, Black *et al.* 1993).

Such probabilistic algorithms have achieved high success rates when applied to part-of-speech (POS) tagging. However, with a baseline of 96–97 per cent accuracy, the amount of data needed to train more accurate models increases exponentially, and it is not clear that training from a hand-corrected 1000–million word corpus would decrease errors dramatically. Therefore, as a complement, and occasionally as an alternative, to probabilistic methods, UCREL increasingly employs **template analysis** techniques, with programs such as JAWS and the rule-based component of CLAWS (described in Fligelstone *et al.* 1996) all using closely related and essentially similar template-based techniques to reduce errors and/or ambiguity. Template-based methods are applied more generally, without a statistical counterpart, in semantic annotation: for example in linking nouns with textually-related adjectives or verbs (see Wilson 1993, Wilson and Rayson 1993).

With further projects developing along similar lines, in order to avoid further duplication of programming effort when implementing template methods, and to develop a flexible system for developing and evaluating rule sets, we decided to build a general purpose rule interpreter or **Template Tagger**. It has already been tried out on the problems for which we have individual tools, but has also found application in new analytical tasks. In due course, deployment of the tagger in new problem areas will enable us to see what further features could usefully be incorporated.

In the following sections, we describe the main characteristics of the

Template Tagger, discuss its development to date, and refer to specific areas of application.

8.2 Framework for a Template Tagger

8.2.1 General aims

Our goal in developing the Template Tagger was to create a general-purpose program which could be used to apply rules for text annotation irrespective of the particular analytical task in question, and in a way which could utilize, and if necessary amend, any existing mark-up in the text input. The program was required for immediate application to a particular multi-level analysis project,[1] but was designed from the outset with the intention of using it for other tasks, as described in Sections 8.4.1–3.

This was achieved by making explicit various aspects of the annotation procedure which are implicit in the task-specific systems referred to above. To illustrate this point, let us consider a simple rule from the contextual pattern rule set (see Section 7.5), more commonly known as the CLAWS idiomlist:

```
a DD231, great DD232, many DD233
```

Without going into too much detail about the underlying system, the basic purpose of this rule is to tag any instance of *a great many* as a complex determiner (DD2),[2] rather than to allow the system to tag each separate word on a word-by-word basis. What is significant for our argument here is the brevity of the rule, made possible by the fact that the system operates within a tightly confined framework of possible input and output tokens, and tagging operations. In this case the system 'knows' that *a great* and *many* are parts of the lexical **input**, and that the DD231, DD232, etc. are candidate POS tags, i.e. **output** tokens, which must in this case *replace* any earlier list of candidate tags in the event of a match.

Slightly different conventions are used with the JAWS system to format a rule used to identify an active, as opposed to passive, use of a past participle:

```
VH*  R*  V*N[PERF]  {by}[3]
```

In this rule, white space, rather than commas, is used as a separator between input tokens; the output token is indicated by the use of square brackets; and the effect, or 'action' of the rule, is not to add anything to a list of candidate tags, but to modify the existing VVN POS-tag in such a way as to mark it as definitely active, rather than ambiguously active-passive, in other words, to replace one POS tag with another (more

informative) one. Note also that in this rule the input tokens are a mixture of wildcarded[4] POS tags (VH*, R*, etc.) and lexical items ('by'), and that in this rule scheme, the use of curly brackets is used to indicate lemmas.[5]

Both these systems are able to use very terse rule formats because they each operate within a confined, albeit different, framework. Although the default tagging operation in each system is subtly different, it is always the same, so needn't be spelled out, and the possible types of input allowed by each system are so confined (in CLAWS' case only words and POS-tags are present, and in JAWS' case, only these plus lemmas), that mere position or use of brackets can suffice to avoid any ambiguities.

But what if we wish to apply not only POS tags, but also semantic tags, or an additional level of grammatical tags or dependency tags? And what if we wish such tags to be available as input against which to match and fire further rules? And what if we don't always wish the same action to be undertaken, but sometimes one action (e.g. 'add tag to list') and sometimes another (e.g. 'overwrite existing tag with new tag'), or yet another, (e.g. 'delete this tag')? And finally, what if we wish to perform all these tasks with a single system, rather than having to create customized software for each task?

In this case, we would need to re-think the rule format cited above, in order to make it quite clear to a generic system

- which parts of the rule specify input and which output
- at what level of input/output the particular tokens are significant
- precisely what action the rule is to perform.

We might represent the JAWS rule as in Figure 8.1. Although making such information explicit leads to less succinct rule formats, it enables us to design a flexible multi-purpose system required for implementing novel annotation regimes.

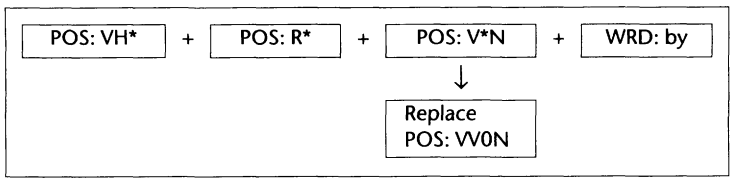


Figure 8.1 Explicit representation within a template rule

8.2.2 Multiple levels of annotation

In the foregoing section we have referred to distinct **levels** of annotation (see Section 1.5), such as 'the POS tag level'. As a general purpose tagger,

the program should support multiple levels of annotation, but we did not wish to pre-judge the range of types of annotation which would be present in the input, or which the tagger would be required to apply. We see the normal operation of the tagger as accepting text with *n* levels of annotation and allowing the user to add one or more levels of annotation to this, building on levels already contained in the input, or simply to alter existing levels of annotation.

The term 'level' is used here to signify a *type* of information. Just as there is no pre-defined set of levels, nor is there any implicit structure or hierarchy of levels. The user is at liberty to use any number of levels, with any significance appropriate to the task and the rules written for it. It would be perfectly possible, if rather pointless, using such a system, to tag all words beginning with 'a' as having the value 'A' at some arbitrary level 'first letter' for which the code FSL has been chosen by the user. The following single rule would suffice.

```
Pattern(WRD: a*/A*)  Action(Inserttag(FSL: A))
```

Assuming that the Template Tagger is able to identify the WRD level in its input stream, the only special information required by the program to apply such a rule is a declaration in its configuration file that the levels WRD and FSL are to be used (see Section 8.2.3).

In its original conception, the Template Tagger allows any number of levels to be declared and used, and rule sets have been developed which manipulate annotations at several levels simultaneously, allowing, for example, grammatical decisions to be based on semantic features and vice versa.

The only obligatory analytical level is the word (or punctuation item) itself (level: WRD). All other levels, and the three-letter codes used to represent them, are introduced at the user's discretion.

For the program to know to which level a particular input token belongs, one of two approaches may be taken. Either all input must be formatted in such a way that each item is explicitly labelled, or the program must be adapted to 'understand' a particular input format. So far, we have taken the latter approach, since we already have well-established text formats in constant use, but the requirement for a particular input 'parser' for any given task is obviously at odds with the goal of a truly generic tool. However, care has been taken to adopt a modular approach to the program construction (see Section 8.3.2), in order to ensure that the heart of the program is universally applicable, with changes only to the input and output modules required to handle new input formats and output requirements. A default output format in which each token of output is labelled for level is also available.

8.2.3 User control

We have already seen that the user must declare which analytical levels are to be handled by the program. Experience has shown that there are a number of other factors which it is useful to be able to vary with rule-driven analyses of the kind the Template Tagger would be used to perform. These include:

1. whether to split the rules into separate files (either for ease of editing, modularity, or selective re-application),
2. in what order to apply the various rule tables,
3. whether to stop searching a rule table once a rule has fired successfully or to continue and apply some heuristic for overlapping items, and
4. how to apply any action specified when a rule fires.

All these choices can be controlled by the user of the system in the **configuration file**, with defaults to enable the novice to achieve simple results without too much knowledge. The configuration file contains two sections, the first listing the levels relevant to the current task, and the second outlining the tagging strategy to be employed.

The only purpose served by the first section in its present form is to provide a check list so that the program may detect when an invalid rule is encountered, i.e. one which contains a reference to an invalid level. In due course the error-checking function of this section could be extended by including names of files containing validation rules for values at the various levels. This could include tag-lists, for closed label sets such as POS tags, or more general format rules for more open-ended annotations. A further type of declaration which belongs in this section concerns the level-specific formatting for the as yet unimplemented 'indexing' function, referred to in the following section.

The second section tells the program where to find the rules and what to do with them; specifically, for each rule file named, how many times to cycle through the rule set before proceeding to the next file, and whether to search to the end of the file in question in all cases ('through' mode), or whether to quit the rule table as soon as a rule is fired ('hit' mode).

8.2.4 A range of tagging operations

The action part of a rule is carried out when a rule fires successfully. Depending on the application, we might need to replace completely the contents of one level (e.g. a complete disambiguation of a set of POS tags) called 'HardInsert', or add a tag or marker to a list of existing values (called 'Append'). 'SoftInsert' adds a tag only if there is currently no value

at the specified level (i.e. it can't overwrite or append). This is useful for ensuring that potentially recursive rules fire only once, such as when identifying the opening of a noun phrase. A function 'Void' has the effect of deleting any existing values at the specified level. We also envisage a 'Remove' function which would remove one value from a list on one level for partial disambiguation purposes, a 'Promote' function which would re-order values stored at a particular level, an 'Index' function which would generate numerical indices for linking items, whether adjacent or non-adjacent, e.g. for anaphoric links or indexed subject-object linking, and functions for modifying existing tagging, e.g. by the addition of a subscript.

One of the most challenging aspects of the development of this software has been to devise and define the minimum range of operations which will cater for most if not all of our tagging requirements.

8.3 Creation of the Tool

8.3.1 Development of the Template Tagger to date

Version 1 of the Template Tagger was written in 1994–95 to apply multiple levels of annotation to text already annotated by CLAWS and subsequently by JAWS (see Fligelstone 1995). Version 1 represents in many ways a prototype, with limited efficiency and some functions as yet unimplemented (see Section 8.2.4), but does adhere to the generic design principles outlined in the previous section.

Later in 1995, the Template Tagger was selected as the tool with which to undertake tag correction and enhancement work on the British National Corpus (BNC).[6] Unfortunately, the scale of the tagging task (some one hundred million words of input, applying several hundred rules) meant that the slow speed of the prototype, which was itself still under development, was too restrictive for the task in question.

Version 2 was thus commissioned; a more streamlined, cut-down version of the prototype. The main concession to efficiency was the dropping of unlimited LEVELS and VALUES. In the prototype, any number of these could be used. In Version 2, levels are hard-coded, and are limited to those relevant to the BNCTE project, and for any token, a maximum of six values may be stored at any given level. Version 2 thus represents a step forward in efficiency but a step back from the goal of a truly generic tool. It is to be hoped that a future Version 3 will marry the virtues of both. All versions have been developed in C on a UNIX platform.

In the following sections we discuss in more detail some features of the

Template Tagger's operations and design, with reference to the issues of flexibility and efficiency.

8.3.2 Input-output modularity

The Template Tagger was initially required to process a single format of corpus, the 'vertical' JAWS format, containing one word or punctuation item per line of input, along with a POS tag (the JAWS tag) and a string representing the lexical headword or lemma. However, the program was always intended to have wider applicability, so input and output modularity was implemented from the start.

The basic concept behind the Template Tagger is that any word-unit in a corpus can be split up into a number of LEVELS corresponding to different facets of information about the word (see Section 1.5). Each level may have one or more values (a familiar example of a multi-value level occurs in CLAWS vertical-format text, where each word may have a number of 'candidate' POS tags, listed in order of likelihood).

The pattern-matching engine within the Template Tagger operates on a sentence (or in later versions, 'unit') level. The Template Tagger reads in a sentence,[7] converts it into an internal data structure, processes it, and outputs the results. The internal data structure for a sentence is generic enough to allow for different formats of corpus. Handling a new corpus style is thus simply a matter of writing new input and output routines, in other words, customizing the tool to be able to assign to the appropriate level each token occurring in the input stream, and formatting the output according to requirements.

Ideally, we would like to create a system which would allow the user to 'explain' to the program how to interpret various input formats, but this is too ambitious at the present time. For now we have confined ourselves to the creation of routines selected by command-line options appropriate to the various formats with which we regularly work. A next step would be to create an input mode which expects all input to be labelled for level. The usefulness of this will be proportional to the extent to which there is agreement and take-up of standard conventions for producing explicitly labelled annotated text (see also Chapter 16 of this volume).

8.3.3 Rule-matching algorithm

The rule-matching algorithm is the heart of the Template Tagger. As has been mentioned, rule matching currently works on one sentence at a time – an attempt is made to match each rule provided by the user at each position in the current sentence, starting with the first word. If several

rule files are to be used, the sentence is processed in its entirety using one file before proceeding to the next file and returning to the start of the sentence. If each rule file contains a different type of rule, this has the effect that the sentence, and by extension, the text, is subjected to one form of analysis before being subjected to another.

Each Template Tagger rule is composed of one or more 'cells', designed to match one or more adjacent items in the sentence. Multiple cells in a rule attempt to match adjacent items in the sentence. Each cell comprises a 'pattern' section and an optional 'action' section. The pattern section details a set of criteria that an input item (i.e. a word or punctuation item plus any associated annotation) must meet for the rule to match. To increase the power of pattern matching, a variant of UNIX-style regular-expressions may be used, including wild cards to represent multiple characters and negation (e.g. POS: NOT a noun).

The action section of the cell is optional, and contains a set of one or more operations to be carried out on the matching word if and only if the rule as a whole fires (i.e. if every cell in it matches). The most basic, and to date the most commonly used operation is 'HardInsert', which erases all current values (if any) at the specified level and replaces them with the value(s) specified in the rule. This operation is useful for adding completely new information, for enriching existing annotation (e.g. replacing general tags with more precise ones), and for tag correction. Actions may be included in any or all of a rule's cells.

An extension to the rule matching system is the optional cell. An optional cell does not have to be matched in order for the rule as a whole to fire. The optional cell takes a number argument, which specifies how many input items it may maximally match. The Template Tagger rule in Figure 8.2 (overleaf) is basically a rendition of the JAWS rule in Figure 8.1, except that the adverb cell (cell 2) is optional, and may match up to three consecutive adverbs. It is possible to include an action within an optional cell, but if the optional cell matches more than one item of input, the same action will be applied to all of them.

The optional cell device produces the problem of possible multiple matches for a rule from a single starting point (i.e. the rule may fire by either matching or omitting to match the optional cell(s)). In such instances, the Template Tagger employs a simple strategy of choosing the longest match. If there are competing match permutations of equal length from the same rule (a rare occurrence, in our experience to date), the Template Tagger will choose the rule-match which matches the cell(s) closest to the beginning of the rule.

The longest match principle also requires that care be exercised in writing rules containing optional cells in a medial position. For example a rule

```
<RULE>
    <NAME> Perfect-1
    <CELL>
        <PATTERN>
            <LEVEL> POS
            <VALUE> VH*
        </PATTERN>
    </CELL>
    <CELL>
        <OPTIONAL> 3
        <PATTERN>
            <LEVEL> POS
            <VALUE> R*
        </PATTERN>
    </CELL>
    <CELL>
        <PATTERN>
            <LEVEL> POS
            <VALUE> V*N
        </PATTERN>
        <ACTION>
            <OPERATION> HardInsert
            <LEVEL> POS
            <VALUE> VV0N
        </ACTION>
    </CELL>
    <CELL>
        <PATTERN>
            <LEVEL> WRD
            <VALUE> by
        </PATTERN>
    </CELL>
</RULE>
```

Figure 8.2 Template Tagger rule with optional cell

intended to capture a noun and the next finite verb, regardless of intervening text, must contain an optional cell to represent that intervening text not as anything at all, but as anything which isn't a finite verb. Otherwise, it is the last finite verb in the sentence which will be matched by the rule, not the first one following the noun. Injudicious use of optional cells can cause whole sentences to be 'swallowed' by rules in this way.

8.3.4 Invisibility

When processing corpora, certain portions of the corpus may interfere

with easy pattern matching. An example would be in spoken discourse where certain aspects of speech (coughs, fillers or pauses) have been transcribed, interfering with the flow of the text, and thus pattern-matching. Invisibility allows the user to instruct the Template Tagger to ignore certain 'words' for the purpose of pattern-matching unless a cell's pattern is explicitly looking for it.

In Version 1, this took the form of a few hard-coded exceptions in the pattern-matching algorithm, notably the CLAWS NULL[8] tag and the CLAWS double-quotes tag. This element was enhanced in Version 2 to allow the user to specify a set of invisible words, based upon their POS values. Ultimately, invisible items should be user-definable with reference to any level or combination of levels.

Future work will focus around improving the TEI-conformant format output module, and increased user-control in the areas of corpus format and output options.

8.4 Areas of Application

8.4.1 Partial syntactic parsing

The first task to which the Template Tagger was put was not the one initially envisaged. Whilst it had been anticipated that the program would first be used for the purpose described in the next section, just as the program was approaching its first trials, Geoffrey and Fanny Leech outlined a set of rules which would produce, on the basis of POS-tagged input, a partial scheme of syntactic labels, not necessarily balanced, which they would then use as input to a further syntactic analysis program designed to complete the parsing task (cf. Garside and Leech 1985).

These rules lent themselves easily to conversion to the Template Tagger format, and the program was successfully deployed on this task. Two levels, LBR and RBR were introduced, to handle left (opening) bracketing

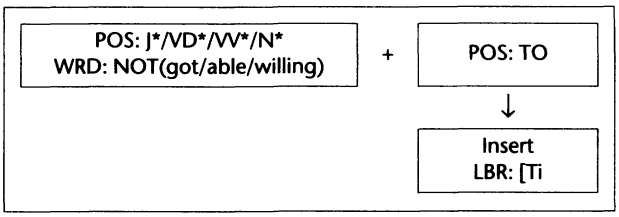


Figure 8.3 Rule to mark opening of an infinitive clause

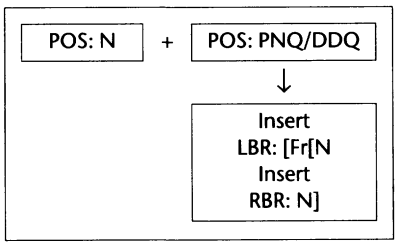


Figure 8.4 Rule to mark opening of a relative clause

and right (closing) bracketing respectively. Figures 8.3 and 8.4 contain examples of the kind of rules produced.

8.4.2 Semantic tagging

The most elaborate use made of the Template Tagger to date has been in the creation of a multi-level annotated corpus with various kinds of grammatical and semantic information added. The progress made did not amount to a complete semantic analysis, as many more rules would have been required to make such a claim, but it none the less demonstrated the value of the program in applying a layer by layer analysis of text, using rules in later stages of the analysis which depended on information added at an earlier stage. All these stages could be incorporated into a single execution of the program so that the input text could be dealt with thoroughly, on a sentence-by sentence basis.

Input to the program was an enriched POS-tagged corpus which incorporated a column showing the lemma or lexeme (which allowed for more succinct rule writing than would have been possible had only the word been available), produced by the program JAWS, one of the earlier template-based annotation programs which in fact led to the development of the Template Tagger (see Sections 8.1 and 8.2 and Fligelstone 1995).

Rule files applied by the Template Tagger dealt with analytical issues such as clause boundary identification, subject identification, main dependencies, phrasal verbs, verb disambiguation, lexical look-up, fixed idioms, stereotypical sentences, and so on. Occasionally a rule would apply several levels of tagging at once (stereotypical sentences being a case in point), so that, for example, semantic information might be applied at a much earlier stage in the process than would normally be the case.

By combining all the levels of analysis into a single all-embracing tagging process, it was all too easy to commit the error of including a rule which could never fire because it depended on patterns which could only

be produced by rules occurring later in the run. To overcome the problem it was found useful to adopt the convention of grouping rule tables into sets which were labelled A, B, C and so on. This was purely a matter of convention with no programming implications. The first few rule files consulted were A-files, followed by the B-files, and so on. Although an A-file rule could be used to apply any level of annotation, the patterns which its rules sought in the text could only contain the kind of information available in the JAWS output. A rule which required information about heads and clause boundaries, largely applied by the A-rules, would therefore have to be included in a B-file, and rules requiring higher level input such as semantic tags would be confined to the C-files, and so on.[9] Figure 8.5 demonstrates the output format and the levels of information encoded.

Ref. No	Pos	Word	Lemma	Clause	Dependent	Head	Function-Semantics
0000182 030	PPMS1	He	HE		<1>	&Nn	SBJ-Human-Male
0000182 040	VABD	was	BE				
0000182 050	VV0G	wanting	WANT		<N	&Va	VRB-Wanting-1
0000182 060	TO	to	TO	+TI			
0000182 070	VV0I	know	KNOW		<V	&Va	VRB-Thinking-Know
0000182 080	CSW	whether	WHETHER	+FN			
0000182 090	PPMS1	he	HE		<2>	&Nn	SBJ-Human-Male
0000182 100	VM00	could	COULD				
0000183 010	VV0I	expect	EXPECT		<N	&Va	VRB-Wanting-5
0000183 020	PPY00	you	YOU		<V	&Nn	OBJ-Human
0000183 030	IF	for	FOR			&P	ADV-
0000183 040	NN1	lunch	LUNCH			&Nn	Concrete-Food
0000183 041	.	.	.				

Figure 8.5 Sample Template Tagger output

8.4.3 British National Corpus enrichment and correction

The Template Tagger was used as an important part of the British National Corpus Tag Enhancement project (BNCTE) described in more detail in Chapter 9. The BNCTE team used Version 2 to apply rules for the assignment of part-of-speech tags that were too complex for the CLAWS tagging formalisms. For this task it was necessary to use only the levels WRD, POS and DEC. DEC is the CLAWS **decision code** – a two figure code indicating which part of the program (lexicon, suffixlist, etc.) had assigned the tag.

It had been found that there were errors in the CLAWS tagging of the BNC that could not be correctly resolved using the CLAWS resources. The Template Tagger, however, was powerful enough to encode a large number of more complex rules. While it was not necessary to use the full range

of functionality of the Template Tagger, at least not in terms of levels, the BNCTE team did have to formulate some very complex rules. A further new problem was how to apply hundreds of rules to a 100 million word corpus while ensuring that the sequence of rules was correct to achieve the desired effect. Careful preparation and testing had to be carried out in order to appreciate fully how the effects of different rules interacted.

For example, rules were written and applied to disambiguate words such as *before* and *after*, which may be tagged as subordinating conjunctions or prepositions, depending on their syntactic role in the sentence. Compare:

We met again after_CJS the ball was over.
We met again after_PRP the ball.

With the Template Tagger it was possible to formulate a rule that looked to the right to see if there was a finite verb within the clause, and if so, tagged the word as a subordinating conjunction. If there was no finite verb before the end of the clause/sentence, then the word was tagged as a preposition. Other rules also ran in conjunction with this, to correct special cases which would not be captured by the main rule, such as *before* occurring at the beginning of a sentence before a specific date like *1965*. It became apparent that it was preferable to run the rules that disambiguated finite verbs from non-finite verbs and nouns before the rules for *before* and *after*, so that the latter rules could properly identify finite verbs in the context.

Without a thorough syntactic parse, it was impossible to correct all errors, but the Template Tagger was crucial in the BNCTE project (see Chapter 9) for improving the accuracy rate of the automatic tagging in areas where the probabilistic formalisms of CLAWS and the restricted power of contextual pattern-matching rules had not been able to make an impression before.

8.5 Conclusion and Further Development

It will be apparent from the foregoing account that we are still some way from the completion of a truly generic template tagging program, but it is encouraging that the three types of deployment discussed in the previous section have all been possible within the framework of the development of a single piece of software.

It remains to be seen whether eventually the general-purpose nature of our tool will be so well developed that it will be possible to bring it to bear on novel tasks without the need for modification. That may be hoping for

too much, but what is clear is that as the Template Tagger matures it will become an increasingly useful analytical and experimental tool. There has already been mention in this chapter of features which are still at the planning stage or under development. As our experience in using the tool grows, so some of those features may be subtly re-defined or supplanted by more pressing concerns, but the following areas seem likely to receive attention:

- **Validation procedures** To date error checking is confined to confirmation that input and rules are well formed, and that there is no reference to spurious levels. With other tools it has been customary to check the content of information levels, e.g. to check for an illegal POS tag. Therefore some means of specifying legal and illegal content at the user-defined levels would be appropriate.
- **Conversion tools** As well as coping with corpus encoding formats, we need to take account of rule file formats that currently exist. For example, in order to save recoding the thousands of rules in the IDIOMTAG module of CLAWS we have automated their translation to the Template Tagger format. This will also aid the acceptance of the Template Tagger if it is to replace our current tools.

What this tool exemplifies is an approach based on the idea that useful labelling of text can be based on the treatment of significant fragments of text, sequences of items which may be specified in templates, without respect to the 'well-formedness' of the broader context. Such approaches promote robustness, as they are more tolerant of 'real language', though their analyses may be less 'neat' than those achieved by more traditional 'structural' parsers. Robust analysers now seem to fall into two distinct types: the probabilistic tagger, of which CLAWS remains an example, and the template based 'fragment' analyser, of which the work on Constraint Grammar (see Karlsson *et al.* 1995) is perhaps the most thoroughly worked out instance to date. The Template Tagger is in the same tradition, though less theoretically oriented, intended for deployment on a range of tasks to be determined by the user, and ultimately as a tool with which to develop new analytical methods.

Notes

1. Lancaster Database of Linguistic Corpora (an ESRC-funded project at Lancaster University, 1990–95). This project involved the creation of a half-million word corpus, drawn from the texts contained in the British National Corpus, annotated to include enriched POS-tagging, grammatical functional labels (Subject, Object, etc.), lexeme identification, principal dependencies, and

some word-level semantic information (ESRC Project No. x205262001).

2. The final two digits in each tag turn the tag DD2 into what are called 'ditto tags': see the discussion of multiwords in Section 2.2 (1).
3. This rule states that given the sequence: any form of *have*; any adverb; any past participle; the word *by*, then the past participle is an active 'perfect tense' participle.
4. A **wild card** (the term is borrowed from the card game Canasta) is a symbol whose function is to stand for **any** value from a range of possible values. Thus, in this case, the asterisk is a wild card symbol which can stand for, or match, any string of characters (including zero characters or one character) excluding a space. Wild cards are extremely useful devices for automated text annotation, in that they allow the use of a **partial** specification, which can match on an open-ended set of **full** specifications.
5. A further distinction which this framework allows is between lemma (or lexeme) and spelling. The use of uppercase within curly brackets would allow a match against any part of a lemma, rather than the exact form cited.
6. This work took place within the British National Corpus Tagging Enhancement project (BNCTE) at Lancaster University, funded by the EPSRC: see Chapter 9 of this volume.
7. The sentence or 'unit' constraint is currently imposed by the rule-matching engine. The input routines can actually be set to handle a sliding window of several sentences using a device known as the 'wheel' devised by M. E. Bryant, formerly of UCREL. The ability to apply rule-matching routines across sentence boundaries would open up the possibility of using the Template Tagger to experiment with rules for anaphoric linking, for example.
8. The CLAWS NULL tag is used to tag apparent words (normally preceded and followed by a space) which are not words in a linguistic sense, such as SGML tags..
9. See the multiple passes through contextual rules in the CLAWS tagger, Section 7.5 above.

9

Improving a Tagger

NICHOLAS SMITH

This chapter explores various strategies that may be employed to enhance the performance of a tagger. It reports on the measures that have been taken to enhance tagging software, and hence to produce a better quality of tagged corpus, in this case the British National Corpus (BNC), in a research project supported by the EPSRC.[1] In this respect it follows on from Chapter 7, but it also describes some developments parallel to those in Chapter 8.

9.1 Meeting more Exacting Standards

Part-of-speech tagging is now a relatively mature NLP activity, in that for some years it has attracted a great deal of research interest and has led, in many cases, to the creation of accurate tagging software. There are a growing number of practical applications where the use of taggers, or of tagged corpora, has been made or at least explored (for example, in lexicography, information retrieval, speech processing, and language learning – see Section 1.2, also van Halteren (forthcoming) I.2). Inevitably as these applications have widened, and research into tagging has intensified, so expectations have risen. Greater demands are being made:

1. It is no longer so impressive, as it once was, to report a 3–5 per cent error rate for tagging English texts; this level of accuracy has been attained for some time (e.g. Marshall 1983, Church 1988) and is now not so much a target as a baseline figure on which to improve.[2]
2. Greater robustness is expected of taggers and of annotation software generally: they should be capable of giving at least some analysis of any input sentence in virtually any kind of text. In many cases the range of text types and domains included in corpora is more adventurous than before. For example, recent large-scale corpora of English such as the

British National Corpus (BNC), the Bank of English (see Chapter 1, n. 2) and the International Corpus of English (Greenbaum 1992) capture many varieties of the language – e.g. spoken impromptu dialogue, teenage pop magazines and unpublished material such as personal letters and student essays, in addition to the more traditional fare of quality newspapers and expository prose articles such as are found in 'first generation corpora' like LOB and Brown. The aim is that accuracy should not dip substantially from one genre to the next.

3. A desirable feature of the modern tagger is that it be able to process texts in more than one language, without requiring special adaptation of design (see Section 10.6).
4. Finally, speed is important too: taggers have to be fast, since they may have to be run in conjunction with other annotation or retrieval software. Also, the corpora that need to be tagged nowadays are often enormous, in comparison with early corpora. Slow programs just will not get the job done.

It is worth noting that some taggers are more specialized, others more general-purpose. At the general-purpose end of the scale there are taggers which aim towards the goal of language-independence (e.g. the Xerox tagger: see Chapter 10, especially Section 10.6), so that the software can be easily adapted to a new language. At the other end of the scale, there are taggers whose application is limited because their resources (e.g. their lexicon, or idiomlist) are tailor-made, or have been trained, for a particular variety or genre of text. Another type of limitation is one of input format: one tagger may be designed for a particular type of text input, whereas another may be easily adaptable for a range of inputs.

This chapter focuses on our own research efforts to deal with two of these greater demands: improving accuracy and adaptability. We describe our attempt to erode the (on average) 3–5 per cent residual errors, and to maintain this accuracy across an exceedingly broad range of text types. The improvements we have achieved have not come through adherence to any one strategy, but rather through a **combination** of strategies. Thus, as was discussed in Chapter 7, the current version of our tagging program CLAWS4 is a **hybrid tagger**, combining elements of the probabilistic and rule-based models. The probabilistic component uses a hidden Markov model (see Section 7.1) and the rule-based component is a contextual pattern matcher (see Section 7.6).

We decided to pursue the hybrid approach further because of both pragmatic and theoretical reasons. Since we already had a robust and well-researched tagger, CLAWS4, it made sense to aim to improve tagger performance within the existing framework, making the most of its

probabilistic component, as well as to its rule-driven component, which even in CLAWS1 had reduced tagging errors by 2–3 per cent.

On the probabilistic side we decided to concentrate on the lexicon, where we felt there was clear room for improvement. On the other hand, our experience was that many types of tagging ambiguity are very difficult to resolve without recourse to linguistic knowledge: hence the **rule-based** component needed enlargement. For both components we wanted to know if the size of the tagsets being used had any effect on the tagger's performance; the availability of a part of the BNC in two tagsets enabled us to investigate this further.

This is not to say that significant improvements could not have been achieved by refinements in other areas of the existing tagging system. We knew, for example, that the morphological 'guesser' component – the method of dealing with unknown words – could have been improved. We might also have implemented a different order of probabilistic model than the existing first order one.[3] However, given a limited time for experimentation, we decided to concentrate on those areas we thought would have the greatest impact on tagging accuracy.

9.2 Achieving Improvements in UCREL Tagging

9.2.1 Creating and exploiting a 'benchmark' corpus

In NLP research it is common practice to have a 'benchmark' corpus for training and testing software. The idea is that such a corpus is divided between a 'training corpus' part and a 'test corpus' part. The former is used for extracting information for developing or improving software. The latter is 'hidden', i.e. never inspected by the researchers, and is used to test how far changes in the software improve or indeed degrade its performance, measured against the **benchmark** of what has been previously achieved. The comparison between the benchmark and what is being currently achieved can be made automatically. In this project, our benchmark corpus was a two-million-word hand-corrected subset of the BNC, known as the Sampler Corpus.

The Sampler consists of a diverse range of text types, broadly sampled from the whole of the BNC, and is divided into spoken and written sections of approximately one million words each. Ninety per cent of each subcorpus is generally used for training, and ten per cent for testing. Because it has been fully hand-corrected and each subcorpus reflects the heterogeneity of the range of texts in the BNC, we have been able to derive from the Sampler better estimates of tag transition probabilities, for tagging the

whole BNC. This has been especially important for tagging spoken texts, as previously CLAWS used transition frequencies derived from written texts only. This improvement in the probabilistic model alone has yielded a 0.3 per cent improvement in tagging.[4] We decided not to extract separate transition matrices from different text types in the Sampler Corpus, as we considered the frequency data to be too sparse:[5] each text type would be represented by no more than 100,000 words at the most. The result has been reasonably accurate tagging overall, rather than particularly good performance in any one genre.

9.2.2 Use of a more fine-grained tagset

When undertaking the original tagging of the BNC, we devised two distinct tagsets, C5 and C7 (on the size of these and other tagsets, see Section 2.3.3). C5 was intended to be a simple, 'basic' tagset of 61 tags (with a simple naming convention) for the benefit of the widest range of users. We also thought that this smaller tagset would lead to reduced automatic tagging errors, since on average fewer choices would need to be made in the disambiguation of each word.

The creation of the Sampler Corpus enabled us to evaluate, in an incomplete but still insightful way, the effect of tagset size on CLAWS's performance. The other, richer tagset, known as C7, resembles more closely than C5 the earlier tagsets (e.g. the LOB tagset) employed with CLAWS. It has 146 tags, and (for example) distinguishes several varieties of preposition, conjunction, noun and personal pronoun not distinguished in C5 (see Appendix III, for a listing of the C7 and C5 tagsets). Figure 9.1 shows some of the many-to-one mappings between C7 and C5.

Experimental results suggest that richer and more informative tagsets yield greater accuracy (see Section 10.4.1; also Elworthy 1995). This was borne out in our own research, where the C7 tagset produced an average

	C7 tagset	**C5 tagset**
Coordinating conjunction (*and*, *or*)	CC	CJC
Coordinating conjunction (*but*)	CCB	CJC
Subordinating conjunction (*after*, *while*, etc.)	CS	CJS
Subordinating conjunction (*as*)	CSA	CJS
Subordinating conjunction (*than*)	CSN	CJS
Subordinating conjunction (*that*)	CST	CJT
Subordinating conjunction (*if*, *whether*)	CSW	CJS

Figure 9.1 Conjunction tags in the CLAWS C7 and C5 tagsets

reduction of error of 0.35 per cent in the spoken and written training corpus. In some cases, the failings of C5 were obviously due to a lack of an important linguistic distinction. For example, a common error was tagging a conjunction such as *before* as a preposition before a subject (nominative) pronoun such as *she*. This very rarely happens in C7 because the subject distinction is explicitly marked on the pronoun where possible– whereas in C5 all personal pronouns are given the blanket tag PNP. On the other hand, errors made in C7 which were not found in C5 tended to involve minor category distinctions, such as the mistagging of *spring* as a temporal noun rather than a general common noun.

The discovery that the larger tagset was less error prone than the smaller one led us, in the original BNC project, to distinguish between a 'process tagset' for internal processing and a 'user tagset' for human consumption – and in this way, to get the best of both worlds. In our recent BNC Tagging Enhancement project, CLAWS4 runs with C7 tags, and the C7 output is finally mapped on to C5 tags, which are the tags output for the user. This means that the greater refinement of the larger tagset is used to obtain (slightly) better results in automatic processing; then the output is mapped onto C5, and in the many-to-one mapping process, some errors again disappear through the process of merging the tags responsible for those errors.

9.2.3 Changes to the lexicon

In very many taggers the lexicon is the main source of initial tag assignments to words in the input text, prior to disambiguation. When a word is 'known' to the lexicon, being listed with its associated tags, the tagging of that word is usually more accurate than when it is 'unknown' and some means of guessing (e.g. on the basis of word endings) has to be relied on.[6] In a probabilistic tagger, moreover, it is highly desirable that the tags for each word have reliable probability values, because of the significant part these play in the disambiguation process.

The earliest CLAWS (CLAWS1) lexicon was small (consisting of about 7,000 entries in the early 1980s, and rising to about 15,000–23,000 entries in various lexicons used in the early 1990s). The rudimentary weighting system for probabilities in the lexicon have already been mentioned; these were sometimes modified after experimentation. When we first tagged spoken data (1993) we made further adjustments to some entries, particularly those capable of acting as discourse markers in speech, so that in effect we had two lexicons (of the same size, and largely identical): one for spoken language, and one for written. The following are example entries from each lexicon, with an explanatory gloss:

though CS RR@ ['*Though* is a subordinating conjunction, or (rarely) an adverb'; from written lexicon]
though RR CS@ ['*Though* is an adverb, or (rarely) a subordinating conjunction'; from spoken lexicon]

To enhance CLAWS's performance we sought to increase the coverage of these lexicons and at the same time to improve on the limited probability information they contained. An attractive solution was to induce (i.e. empirically derive) a lexicon automatically from the Sampler Corpus. After extracting a frequency list of the words in the corpus, we used the relative frequencies of the respective tags as providing estimates of tag probabilities for each word. The induced numerical probabilities replaced the crude symbolic markers previously used, providing the statistical base for a thorough-going HMM. The following shows the contrast between the hand-crafted lexicon and the induced lexicon for the word *clean*, using C7 tags:[7]

clean JJ VV0 RR % NN1 %
clean JJ 66 JJ: 1 VV0 42 NN1 2 RR 2

In theory, everything seemed in favour of substituting the induced quantitative lexicon for the hand-crafted lexicon with subjective frequency weightings. The induced frequencies were 'natural' and 'real', rather than concocted; and we would obtain a much larger lexicon than we already had (one in the region of 45,000 entries for written English, and 20,000 entries for spoken English). However, in practice, the results were far from unequivocally in favour of the induced lexicon.

For our experiment, we used two training corpora of 900,000 words, one for written and one for spoken English, and two corresponding test corpora of 100,000 words. Comparative results are given in Figures 9.2 and 9.3. (Error rates are for spoken and written texts by word frequency in training data, and comparison is restricted to words listed in both induced and handcrafted lexicons.)

In both the spoken and written trials we found that, whichever lexicon is used, on average tagging accuracy increases with the frequency of the word in the training corpus. In the written experiment, the induced word entries start to yield better results than the handcrafted equivalents when at least 50 tokens occur in training. Presumably this is because the word's distributional behaviour (i.e. the relative probabilities of each word class tag) grows more evident with frequency – or, to put it negatively, the more frequent a word-tag pairing is, the less it is likely to be influenced by biasing factors of chance. Yet the results are far less clear in the spoken experiment, where (for reasons which remain unclear) the induced

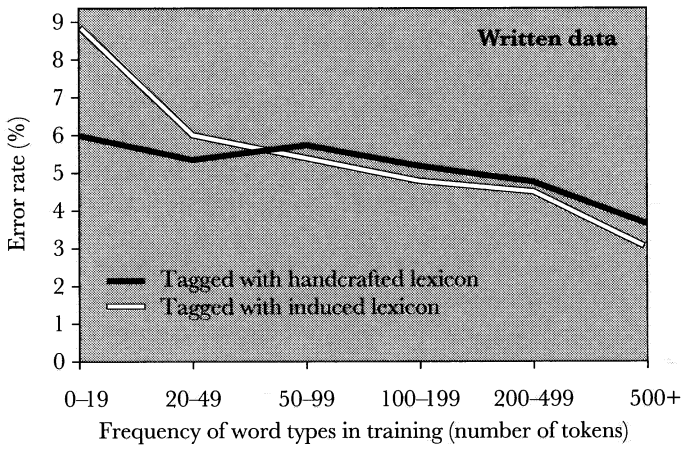


Figure 9.2 Comparison of tagging performance when using automatically-induced and handcrafted lexicons (written data)

lexicon entries start to outperform their handcrafted counterparts only at frequencies of 500 or more.

What we learned is that surprisingly good results are obtained from the apparently crude handcrafted probabilities. Even some high frequency words are better tagged with the handcrafted lexicon. The outcome of this experiment was that we have created two merged lexicons (one for

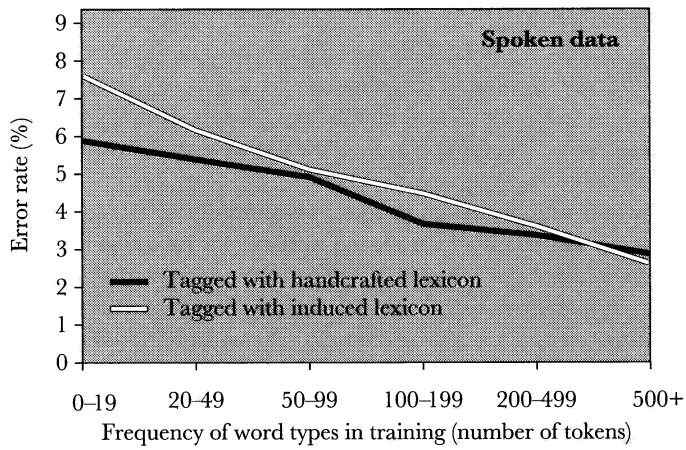


Figure 9.3 Comparison of tagging performance when using automatically-induced and handcrafted lexicons (spoken data)

spoken, the other for written data), consisting of a mixture of handcrafted and induced entries. The handcrafted entry remains the default, but entries from the induced lexicon are substituted for the handcrafted ones only where the former give a 5 per cent improvement on the tagging of that word (as evaluated on the test corpora), and where there are at least 50 occurrences of the word in the training corpus. In this way, we arrive at a lexicon which is more successful than either the handcrafted or the induced lexicon alone.

Lexicon inducing has not been the rich source of new entries that we had hoped. One reason for this is that most of the entries in an induced lexicon represent only one or two occurrences of the word in question. There is a particular problem with rare ambiguous words (e.g. *braised* may be past tense, past participle or adjective) for which one or more of the possible tags are quite likely to have zero occurrences, and therefore not to appear in the induced lexicon. Our main method of increasing lexical coverage has been more labour-intensive: it has involved searching through the BNC for words unknown to the current lexicon. This can be done by an automatic scan of the corpus, and the list of words so found, with their immediate contexts, can be inspected to identify words which have been mistagged. The word is then inserted in the lexicon, of course with its correct list of tags.

9.2.4 The rule-based component: use of context-sensitive rules

Use of linguistic rules now constitutes a substantial part of our overall tagging procedures. This has been motivated by observation that tagging ambiguities are often difficult to resolve without taking account of a larger context than a typical HMM can allow for.

Rules have two principal points of insertion in our tagging process: either immediately before or immediately after the statistical disambiguation phase of CLAWS4 (see Section 7.6). Patterns in the 'before' rules generally consist of fixed expressions and templates (*of course, New York, as far as...concerned*) and their associated tags, or partially resolved patterns (e.g. *in general, kind of* – which can either take ordinary tags assigned by the lexicon, or be treated as adverbial units). Such rules are a means of pre-empting errors by the probabilistic component in dealing with idiosyncratic word sequences.

The 'after' rules tend to be more context-sensitive and may require some disambiguation to have been done either probabilistically or by rules applied earlier. Their main function is to correct errors in the output from

CLAWS4, and for this reason they can be likened (borrowing from Brill's 1992 terminology) to 'patching rules'. An example of such a patch (using C7 tags) is:

```
[VM], ([XX/RR])2, [VV0] VVI
```

which is interpreted: 'If there is a sequence of the following kind:

(a) a modal auxiliary
(b) [optionally] up to two words which are adverbs and/or negative particles
(c) a finite present tense base form of the verb

change (c) to an infinitive verb.' It would result in changing the word *succeed* in *may not always succeed* from a finite base form to an infinitive. Such rules have now grown to some 4,000 entries, most of them sorted for convenience into categories such as foreign expressions, naming expressions, noun compounds, and phrasal verbs – these being lexico-grammatical phenomena of English which often require such pattern-matching rules. However, there is also a general class of rules, illustrated by the example above, which in fact capture local syntactic constraints. Yet even after such rules had operated in CLAWS, many errors remained, suggesting that we needed (i) a more powerful pattern-matching rule formalism to resolve more complex errors, and (ii) a more systematic analysis of errors, to determine what rules would be required.

9.2.5 A corpus 'patcher'

The solution to need (i) above was to adapt the general model of the Template Tagger discussed in Chapter 8 (see especially Section 8.4.3). That is, we used the functionality of the Template Tagger, with its powerful pattern-matching capabilities, but only insofar as these related to our present concern with the BNC, which was the manipulation of words and tags as in the 'after' type rules discussed above. Our adapted version of the Template Tagger will be termed a **patching tool** (or 'patcher'), since in many ways its function is comparable to the patching mechanism of Brill's tagger (see Section 7.1), in taking as input a faultily-tagged corpus, and outputting the same corpus with the tagging improved. One advantage of this tool over the rule-based component of CLAWS4 is that the patcher has a wider context window, extending up to sentence boundaries, so that longer-distance constraints can be captured. Another advantage is that it has a definition facility which enables the analyst to define categories or sets of words in addition to the categories implicit in the tagset. For example, a category declaration `#ADVERB` is a convenient was of referring to all

adverbs except adverbial particles tagged RP; #COLOUR represents any declared colour, and #PUNC1 any punctuation marking the boundary of a clause. This supports a general principle, stated in Tapanainen and Voutilainen (1994), that in knowledge-based disambiguation it is desirable to have as much information and functionality as possible. To give a further example: the output from CLAWS which is input to the patching tool is the 'vertical output' (illustrated in Section 7.6) containing not only the preferred tag, but also other tags of lower likelihood. It is then possible to refer to these less likely tags in the rules for the patching tool. e.g. [JJ~VV0] means 'where the CLAWS-preferred tag is JJ, but VV0 is also output as a possible tag'.

Our approach to need (ii) at the end of the previous section, the systematic analysis of tagging errors, has been enhanced by the method of 'parallel concordancing'.[8] This consists in aligning the latest CLAWS4-tagged version of the training corpus with the benchmark ('correct') version, and generating a concordance line for each instance where the former assigns an incorrect tag *A* and the latter a correct tag *B*. In the brief concordance excerpt shown in Figure 9.4, *A* is a subordinating conjunction (CS) and *B* a preposition (II).

ing the company which have occurred	since CS [II]	the balance sheet date . **11;7898;ptr **1
lmond-green shirt with epaulettes .	Before CS [II]	the show , the uniforms were approved by
heart towards the library catalogue	since CS [II]	the advent of online systems . The overall
n sales . There have been no events	since CS [II]	the balance sheet date which materially af
been in demand , adding 13p to 173p	since CS [II]	the end of October . Printing group Linx h
of Hugh Candidus of Peterborough .	After CS [II]	the appointment of Henry of Poitou , a sel
my boys would be in the Ravenna mud	until CS [II]	the spring . Our landlady obviously liked
volution in treatment brought about	since CS [II]	the arrival of penicillin and antibiotics

Figure 9.4 Parallel concordance indicating conjunction-preposition tagging errors

The linguist works interactively with the concordance and applies linguistic knowledge to discover the patch rules.[9] For example, as Figure 9.4 shows, many words mistagged CS lack a finite verb in their right hand context (that is, they do not mark the start of a clause). On this basis one might propose a tentative rule to correct the tag to II:

```
[CS~II] II, ([!#FINITE_VB])inf, [#PUNC1]
```

Interpretation: 'If a sequence of the following kind occurs:

a. a word tagged as a subordinating conjunction but which could be a preposition
b. an interval of any number of words, provided that none of them are tagged as finite verbs

c. a 'hard' punctuation boundary (./;/:/?/!)

then change the conjunction to preposition.'

The linguist can sort the concordance and apply constraints in order to home in on the precise form of 'template' needed to correct a certain set of errors, while minimizing the set of new errors created by the rule.

After a preliminary set of rules has been developed, they are evaluated by being run over the CLAWS-tagged training corpus and a fresh parallel concordance is generated. This cycle is repeated until the errors for that tag pair are substantially reduced, and the rules can then be applied to the test corpus, to confirm their efficacy. At present nearly all patch rules are worked out manually. Only a few very short-range rules, derived from frequently mistagged word sequences, are extracted automatically. In our experience, creating patches is more of an art than a science – although the patches may then be scientifically checked against the test corpus.

The ordering of the rules – as in Brill's tagger – is important. We find it safest and most practical to run the patches over the corpus in several different passes, rather than in one single pass. Experience shows that it is better to play safe, usually putting short-range rules in an early pass. This aids more complex disambiguation in subsequent passes. In each pass, the more specific rules are placed earlier, so that if there is conflict with a more general rule later, the specific rule wins, and the more general rule becomes a default. Rule ordering is a little more complex than this – for example, many rulesets can be fired in any order, but for complex function words like *as*, rule ordering is crucial.

9.3 Improving the Output and Measuring Accuracy/ Ambiguity

In spite of the running of a large number of patches over the BNC, there still remains a substantial residuum of error – in the region of 2 per cent – in the corpus. This well illustrates the diminishing returns of any effort to achieve the accurate tagging of a corpus of any kind, but particularly of a corpus of such varied language – spoken and written – as the BNC. Ultimately, human language use is so intractable and so full of idiosyncratic phenomena that even a set of complex and well-targeted patches may only succeed in eradicating about half the errors they need to deal with.

Given that large tagged corpora (where hand-correction has not been done) are likely to be error prone in the foreseeable future, we have had to consider what is the most useful form of output for the user. One obvious possibility is to output just the 'winning' tag per word token, in which

case the user has to take note of the fact that some tags (say, 2 million in a corpus of 100 million!) are incorrect. Another possibility is to output the CLAWS 'vertical' format as in Section 7.6., where the alternative tags are listed alongside the CLAWS-preferred tag. A third alternative is one we chose for the BNC, both for the original tagging of the BNC (1991–95), and or the enhanced tagging we have undertaken more recently (1995–96). The method is to produce a 'horizontal' output in which most tags are disambiguated, but where a small percentage of tags (in the original BNC about 5 per cent) are output as **ambiguity tags** (also called **portmanteau tags**) representing two alternatives.[10] These ambiguity tags have labels such as VVD-VVN, which indicate that the correct tag is one or the other of two tags (in this case VVD 'past tense' or VVN 'past participle'). The decision as to whether to output ambiguity tags is made by CLAWS4, according to a probability threshold which is calculated for each relevant tag-pair. That is, if the difference between the tag with the highest probability and the tag with the lower probability is less than x (x being tailored to the individual tag-pair), then the ambiguity tag is output. Otherwise, it is assumed that CLAWS4 'is sure enough to decide' that one tag is the correct one.[11]

With the ambiguity tag output, ambiguity is traded for error. It is a matter of judgement whether this is a good bargain; but we have felt that an output in which a smaller number of errors occur, but in which a relatively small quantity of ambiguities occur, is better for the average user. A bonus, for this approach, is that many of the ambiguities represent 'grey area' cases where, in fact, human linguists would hesitate whether to consider a word a member of one category or the other. This is commonly the case, for instance, with the ambiguity tags AJ0-NN1 'adjective or [singular] common noun' or NN1-NP0 '[singular] common noun or proper noun'. Our current result, extrapolating from applying ambiguity tags to the test corpus, is that the output form of the corpus will have 1.18 per cent error and 4.6 per cent ambiguity tags. This may be alternatively expressed (see Chapter 7, n. 1) as Recall: 98.82 per cent, Precision: 95.4 per cent. It is worth noting that had we not employed ambiguity tags (simply taking the CLAWS-preferred tag in all instances) the error rate would have been 1.99 per cent. That is, ambiguity tags have 'cut' our outstanding errors by approximately 40 per cent.[12]

9.4 Conclusion

This is obviously not the end of the story of how to achieve better tagging. For example, one thing we have not yet tried with the BNC, simply for lack

of time, is the tuning of the tagger to each major variety of text in the corpus. We have, to some extent, tailored our resources differently for the written and the spoken subcorpora. But one could go much further than that in recognizing that, optimally, each new text type will benefit from a tailor-made lexicon, tag transition matrix, and rule-based component. How to acquire these new resources for new text types is a problem for the future.

Further, greater recourse to higher levels of syntax seems to be a means of resolving some tagging ambiguities. The Helsinki constraint grammar ENGCG (Karlsson *et al.* 1995) in this respect outdoes other taggers: its rules have greater expressive power, using clause boundary information, its lexicon contains verb subcategorization information, and it can recognize noun phrases. The UCREL patching tool falls short in these respects. The general message is that taggers, like other NLP software, need to maximize information in a knowledge-based system, as well as maximizing the retrieval of information from corpora through statistical and other extraction techniques. This is a powerful argument for the adoption of hybrid tagging techniques.

Notes

1. EPSRC (=Engineering and Physical Sciences Research Council) Research Grant GR/K50054 awarded to UCREL, Lancaster University, 1995–96. Collaborators on the project were Oxford University Press, Addison Wesley Longman, and Larousse. As a result of this work, an improved version of the BNC will be made available, through Oxford University Computing Services, in 1997.
2. Is it possible to reach 100 per cent accuracy? Probably this figure is an unrealistic goal, not only because of software limitations, but because accuracy presupposes some absolutely consistent standard of 'what is correct' (see Section 2.5 and Chapter 17; also Church 1992, Voutilainen 1995 and Källgren 1996). Nevertheless by far the majority of errors made by CLAWS are not controversial.
3. After one experiment which used a trigram rather than bigram HMM, we abandoned this aspect of probabilistic enhancement, since it did not lead to a reduction in tagging errors. As far we we are aware from others' research, implementations of higher *n*-gram models have made no substantial improvement over a bigram model.
4. Because of the extra expense of collecting, processing, and transcribing spoken data, the ratio of spoken to written data in the full BNC is 1:9; in the much smaller Sampler Corpus it is 1:1.
5. On the other hand, much larger quantities of textual data were available for tag transition extraction from the whole 100-million-word BNC. However,

here the transition statistics would be 'noisy' because of the existing tag error rate of 3–5 per cent.

6. See Section 7.4 on the sequence of steps taken in CLAWS4 in the assignment of potential tags to words.
7. Key: JJ=adjective, RR=adverb, NN1=singular common noun, JJ:=adjective spelt with a word-initial capital, VV0=finite base form of lexical verb; %=very rare.
8. We acknowledge the assistance of Paul Rayson for help in developing a tool for parallel concordancing. In addition the powerful query syntax of the Xkwic software (Christ 1994) has proved invaluable for testing rules on the Sampler Corpus and the larger BNC.
9. We considered applying Brill's method (1992, 1994) – see Section 7.1 – for discovering and ordering patching rules automatically, but decided this was too risky and difficult, given the size of the BNC, and given the subtlety of the rules needed. It should be borne in mind that, unlike Brill, we were faced with the task of correcting residual errors after a 'standard' tagging has produced a 95–97 per cent accuracy rate, and were therefore trying to deal with the most recalcitrant categories of errors.
10. Ambiguity tags could in principle represent more than two alternatives, but in practice we have limited the number of alternatives to two, and the number of ambiguity tags to a limited number which account for the most common error types. Further information about ambiguity tags is given in Leech *et al.* (1993).
11. Experiments in outputting ambiguous tags in relation to a statistical threshold are reported also in de Marcken (1990) and Weischedel *et al.* (1993); their thresholds differ from ours in that they are set uniformly for all ambiguous readings, rather than tuned to individual tag pairs.
12. It is also worth noting that if we restrict the calculation to major word class only, merging distinctions such as that between common and proper nouns, recall improves to 99.11 per cent and precision to 96.29 per cent. Accuracy, after eliminating ambiguities, rises to 98.32 per cent.

10

Retargeting a Tagger

FERNANDO SÁNCHEZ LEÓN
and AMALIO F. NIETO SERRANO

10.1 Introduction

Most corpus software applications, and in fact most of the work on corpus linguistics, have been developed for English, and taggers are not an exception. However, since extensive corpus work for other languages is now rapidly emerging, a port of tools and techniques successfully tested for English is needed. If we make an oversimplifying division of tagging techniques into probabilistic and rule-based (see Chapters 7 and 9), the former are the most natural candidates for a test in other languages. Rule-based taggers sometimes include *ad hoc* rule machinery that may prove inadequate or insufficient for other languages.

This chapter describes our experience retargeting the Xerox Tagger, a public domain probabilistic tagger originally developed for English, to apply to Spanish. The goal of this retargeting, performed within the project CRATER (Corpus Resources And Terminology ExtRaction).[1] was threefold: first, to serve the practical purpose of tagging the Spanish version of the International Telecommunications Union (ITU) corpus (see Section 15.1); second, to test the language independence of the software package, as is claimed by the authors; and finally (and most importantly) to try to shed some light on the debate about the language independence of probabilistic techniques themselves.

10.2 The Xerox Tagger

As already mentioned, the Xerox Tagger (Cutting *et al.* 1992; Cutting and Pedersen 1993) uses a statistical method for text tagging. In such systems, ambiguity in the assignment of a tag to a word is resolved on the basis of most likely interpretation. A form of Markov model is used that assumes that a word depends probabilistically on just its part-of-speech, which in

turn depends solely on the category of the preceding words (though only one word is used in the Xerox Tagger).

Two types of training have been used with this model. The first one makes use of a tagged training corpus. A small amount of text is manually tagged and used to train a partially accurate model. This model is then used to tag more text; the tags are manually corrected and subsequently used to retrain the model. This training method has been called *bootstrapping* (Derouault and Merialdo 1986).

The second method does not require a tagged training corpus. The model is then called a **hidden Markov model** (HMM – see Section 7.1), as state transitions cannot be determined, though the sequence of outputs is known. Jelinek (1985) uses this method for training a text tagger. A trigram (three-word-sequence) approach is generally used, where trigram estimates are smoothed out using the method of **deleted interpolation** in which weighted estimates are taken from second- and first-order models and a uniform probability distribution. Kupiec (1989) uses word equivalence classes based on parts of speech to pool data from individual words. The most common words are still stored in a lexicon file, while all other words are represented according to the set of possible categories they can assume. The number of equivalence classes (referred to as **ambiguity classes** in Cutting *et al.* 1992) can be considerably reduced (to approx. 400 for the whole vocabulary contained in the Brown Corpus). As a further reduction of the number of parameters, a first-order (bigram) model can be employed. In these models, a word depends on its part-of-speech category, which depends solely on the category of the preceding word.

The Xerox Tagger is based on an HMM. It uses ambiguity classes and a first-order model to reduce the number of parameters to be estimated without significant reduction in accuracy. According to the authors, reasonable results can be produced training on as few as 3,000 sentences.

The tagger employs the common sequential procedure of **tokenization**, **lexicon look-up** and **disambiguation**. Every token is passed to the lexicon where it is converted into a set of stems, each annotated with a part-of-speech tag. A set of tags ambiguously assigned to a token identifies an *ambiguity class*. The disambiguation is performed by computing the path of maximal probability through a previously trained HMM.

Words not found in the lexicon are assigned an ambiguity class guessed from the lexicon itself and a training corpus, using a function that also computes 'suffixes' (in fact, final sets of characters not necessarily constituting suffixes in the linguistic sense) for these unknown words. As a final mechanism, there is a default ambiguity class for those words neither found in the lexicon nor ending in a recognized suffix.

10.3 A Mixed Model

While the original tagger relies on a function to compute both a set of suffixes and the ambiguity class to be assigned to each of them, this methodology was considered inefficient for inflectional languages like Spanish. In fact, this function assigns only ambiguity classes which are already observed in the lexicon, and are thus validated because of the existence of a unique wordform that ambiguously receives them. However, for suffixes (for instance, for inflectional morphemes in Spanish), the range of tags may be potentially higher than is observable in the lexicon. Therefore, if we consider *a* as one of the suffixes computed (being a common inflectional morpheme in Spanish), it will be impossible for it to receive an ambiguity class that precisely reflects its ambiguous morphosyntactic properties, since there does not exist a unique word, ending in *a* or in whatever other suffix, that has already validated such an ambiguity class. As a consequence, the system is forced to assign the (more ambiguous) default ambiguity class, thus impoverishing the tagging output.

The use of a very fine-grained tagset (see below) may be responsible for this lack of enough ambiguity classes to find the correct one for *a*, since ambiguous wordforms are greatly specialized in their tag assignment. With a smaller tagset, identifying only category distinctions (with no subcategory information), ambiguity classes would have a lower number of terms (arity) and this would mean that the lexicon would accommodate all ambiguity classes needed. However, and most importantly, unlike isolating languages, inflectional languages show the characteristic of having a clear correspondence (in a vast number of cases) between (linguistically motivated) suffixes and morphosyntactic properties of the word(s) they are attached to. Consequently, this *a priori* knowledge should be exploited by the tagging system. Thus, if we *know* that a word ending in *a* can represent the following ambiguity class,

"a" #(: ADJGFS : NCFS : VLPI3S : VLPS1S : VLPS3S)[2]

the system should be able to use this information *without needing to estimate it*, as it does with entries in the lexicon. These entries contain the tag (or set of tags) not possibly but mandatorily assignable to each entry, so there is no need to approximate it.

On the other hand, the practice of manual coding of information for unknown words has been used only to a relative extent in probabilistic models of language. Some systems, like the Xerox Tagger, compute probabilistically both the suffixes and the ambiguity classes associated with them; but others, like the one described in Weischedel *et al.* (1993) include

a hybrid approach where suffixes are manually added and ambiguity classes are approximated directly from training data.

Nevertheless, all probabilistic taggers work with manually coded information such as, for instance, a lexicon, as already mentioned. Hence, the approach proposed could include both manually-computed suffix tables and ambiguity classes, especially for inflectional languages where this information can be straightforwardly obtained, thus improving system accuracy. This approach, however, has the drawback that migrating the system to a new tagset entails more resource conversion work, since both the lexicon and the suffix table will have to be mapped onto it.

This strategy has been successfully implemented in the porting of the original Xerox Tagger to Spanish. It consists in merging, during normal training, the set of classes observed in the lexicon with those stated by a linguist in the suffix file.[3] The training process will benefit from the reduction in the number of elements of the ambiguity classes to be computed when words not contained in the lexicon are found, thus improving accuracy in the generation of paths.

10.4 Model Tuning

Parameter estimation is a central issue in probabilistic models of language. A HMM of language can be tuned in a variety of ways. Thus, several decisions have been taken concerning the tagset, the lexicon, and the **biases**. These choices are presented and (hopefully) justified below. The selection of the training corpus is also discussed.

10.4.1 The tagset

Tagsets used by taggers for English have usually been derived in some way from that used in the Brown Corpus (Francis and Kučera 1982), which distinguishes 77 tags. The trend since the design of this tagset has been to refine and elaborate it. Thus, the Lancaster-Oslo/Bergen (LOB) Corpus distinguishes about 132 tags, and the Lancaster UCREL group uses a set of 166 tags (for CLAWS2–Garside *et al.* 1987). Other tagsets are even larger, such as the one used in the London-Lund Corpus of Spoken English, which contains 197 tags.

These further refinements to the original Brown tagset reflect the necessity for a tagged corpus to show all the (morpho-)syntactic idiosyncrasies of a language. Thus, the rationale behind developing large, richly articulated tagsets is to approach 'the ideal of providing distinct codings for all classes of words having distinct grammatical behaviour' (Sampson 1987c).

On the other hand, some projects based on a stochastic orientation have modified the original Brown Corpus tagset by paring it down rather than extending it. This is the case for the Penn Treebank Project, that uses 35 POS tags (Marcus and Santorini 1992). The decision was founded not only on the use of a probabilistic model but also on the fact that the goal was to parse the corpus, thus some POS distinctions were recoverable with reference to syntactic structure.

However, international initiatives on corpus annotation standards, such as those proposed by EAGLES (Leech and Wilson 1994), recommend the distinction of major morphosyntactic categories within tagsets. In fact, level 1 (L1), including *recommended attributes/values*, distinguishes, among others, *type*, *gender*, *number*, *case*, *person*, *tense*, *mood*, and *finiteness*. EAGLES recommendations explicitly state that '[t]he standard requirement for these *recommended* attributes/values is that, if they occur in a particular language, then it is advisable that the tagset of that language should encode them' (Leech and Wilson 1994).

Supposedly, in the construction of a tagset to be used by a probabilistic tagger, a trade-off must be found between exhaustivity and accuracy: it is assumed that the more exhaustive the information encoded in the tagset (the larger the tagset), the less accurate the tagging will be (since the resulting model will be more complex and parameter estimations less accurate). In fact, our findings suggested otherwise (see below and Section 9.2.2).

This trade-off was taken into account in the creation of the tagset for Spanish to be used by the Xerox Tagger within this project. In a first attempt, a quasi-**ideal** tagset was built, taking into account not only EAGLES recommendations but also TEI guidelines on text annotation (Langendoen and Fahmy 1991, Simons 1991, TEI AI1W2 1991). This tagset, designed using **external criteria**, as described in Elworthy (1995), although a year before Elworthy's paper was presented, is described in Sánchez León (1995). It contains 475 linguistic POS tags. Thus, it is a very comprehensive tagset, distinguishing almost all morphosyntactic features recommended by the above-mentioned initiatives. Information considered includes common/proper distinction for nouns, with various subtypes for proper nouns, some semantic information such as measurement nouns and special tags for days of the week and names of months, gender and number; degree, wh information, locative (with subtypes), deixis, and polarity for adverbs; status (main/auxiliaries), person, number, tense, mood, gender, and finiteness (implicit) for verbs (given the richness of the verbal morphology in Spanish, verbal tags account for 59 per cent of the total number of tags).

This tagset has been considered 'too finegrained to be suitable for a probabilistic tagger', by some researchers [Lauri Karttunen, personal

communication]. At the beginning, it was viewed more like a 'meta-tagset' from which several more restricted tagsets could be mapped in order to find the tagset that maximizes the accuracy of tagging. However, in the trade-off between granularity and accuracy, it was our intention to give priority to the capability of making certain linguistic distinctions over the yet-to-be-proved possibility of improving accuracy by means of reducing the tagset size. In this respect, as has been recently proved by Elworthy (1995), larger tagsets can obtain even better accuracy rates than smaller ones.

Besides this tagset (henceforth *tagset 0*), five other tagsets were derived in order to observe whether the accuracy increases in every subsequent reduction. These five tagsets will be referred as *tagset 1* to *tagset 5*.

Note in the tagset descriptions below that reducing the tagset size does not necessarily decrease the ambiguity rate in the lexicon,[4] and similar figures can be found for every tagset. In fact, the reduction in the number of tags may, sometimes, introduce new ambiguities.

Tagset 0 had 475 linguistic tags. The ambiguity rate for this tagset in the lexicon was 1.1869 tags per word. Information removed from this original tagset in every subsequent mapping is shown below:

- *Tagset 1* Semantic information is removed (deixis for demonstratives, semantic information for nouns, semantic information for quantifiers, units of measurement changed into common nouns), although *wh* information is kept. Degree is also removed from adverbs and adjectives, and polarity from adverbs, prepositions and quantifiers. Negatives are converted into adverbs. This tagset has 387 linguistic tags. The ambiguity rate in the lexicon is 1.1876 tags per word.
- *Tagset 2* Tagset 1 reduction plus: Type of verb distinction is reduced to lexical vs. non-lexical (including all auxiliaries and modals). *Wh* information is removed. Possessive information in relatives is also removed. This tagset has 223 linguistic tags. The ambiguity rate in the lexicon is 1.1876 tags per word.
- *Tagset 3* Tagset 2 reduction plus: Functional information eliminated (pronominal/non-pronominal distinction, case for pronouns). The type-of-conjunction distinction is removed. Only one type for verb is distinguished. Negatives changed into adverbs. This tagset has 152 linguistic tags. The ambiguity rate in the lexicon is 1.1870 tags per word.
- *Tagset 4* Tagset 3 reduction plus: Some morphosyntactic categories are eliminated – tense and mood in verbal tags. Definiteness vs. indefiniteness removed in articles. ACRNM, ALFx, TRATx, ITJN, ROMAN (but not foreign words, PE) tagged as SYMBOL. Portmanteau words (PAL, PDEL)

converted into prepositions. Person distinction removed from pronouns. This tagset has 76 linguistic tags. The ambiguity rate in the lexicon is 1.1823 tags per word.

- *Tagset 5* Tagset 4 reduction plus: Gender distinction is removed but number is kept. Personal/non-personal pronoun distinction is removed. This tagset has 40 linguistic tags. The ambiguity rate in the lexicon is 1.1162 tags per word.

10.4.2 The lexicon

All probabilistic taggers make use of a lexicon of varied coverage. Cutting *et al.* (1992), for instance, report on tagging results on even numbered sentences of the Brown Corpus using a lexicon of 50,000 forms. With this lexicon and the suffix file, no unknown forms were encountered in the training process, thus providing no training data for forms assigned to the open class.

However, a larger lexicon does not necessarily guarantee a better tagging accuracy. Words are usually ambiguous and may take, depending on the context, a different POS tag. The probability of a given word taking one or the other tag may not be the same, though, and some systems have the possible tags for a word arranged in decreasing likelihood, and also include special mechanisms to express the fact that certain tags are 'rare' or 'very rare' (see Section 7.5). When this selection is impossible in the system, other devices may be employed to reduce ambiguity. Some authors use an optimal dictionary that indicates, for each word, all the tags assigned to it somewhere in the corpus being used, but not other, possible tags (Merialdo 1994), but this only makes sense when the corpus is already (correctly) tagged and it is used to measure the accuracy of a tagger for that particular corpus. Others propose the exclusion of rare readings from the lexicon to prevent the tagger from selecting them (Tapanainen and Voutilainen 1994).

Since our starting point is not a tagged corpus on which to perform the testing of a given stochastic model, our lexicon is not specially biased, in terms of tags, towards the corpus we aim at tagging. On the contrary, we would like to build the tagger on as uniform lexical material as possible. Hence, the whole set of tags for each word has been taken into account during lexicon building, with the exceptions mentioned in note 4.

10.4.3 Training a hidden Markov model

Training on hidden Markov models of language is performed without a tagged corpus. In a tagger under this regime, state transitions (i.e. transi-

tions between categories) are unobservable. Under these circumstances, the training is performed according to a Maximum Likelihood (ML) principle, using the Forward-Backward (FB) or Baum-Welch algorithm. This training process can be biased in a number of ways in order to 'force' somehow the learning process. Two such ways implemented in the Xerox Tagger, making use of ambiguity classes and state transitions, are described below:

1. The biasing facts on ambiguity classes are called **symbol biases**. These represent a kind of lexical probability for given equivalence classes. In this way, ambiguity classes are annotated with favoured tags. Note, however, that this is stated for a given class and not for individual forms in the lexicon (as it is, for instance, in CLAWS–Garside *et al.* 1987), resulting in a less efficient mechanism.
2. The biasing facts on state transitions are called **transition biases**. These specify that it is likely or unlikely that a tag is followed by some specific tag(s). The biasing can be formulated either as favoured or as disfavoured probabilities. Disfavoured probabilities receive a small constant but are not disallowed; on the contrary, data in the training corpus may modify probabilities.

Tapanainen and Voutilainen (1994), who use the Xerox Tagger in combination with ENGCG for tagging English texts, reporting an accuracy of 98.5 per cent, propose other ways of tuning the system. These are the following:

(a) Not including rare readings in the lexicon so that the tagger will not select them.
(b) Using different values for the number of iterations (the number of times the same block is used in training) and the size of the block of text used for training.
(c) Choosing corpora of different sizes for training.

In our case, it has already been pointed out that an *a priori* decision was to test the system with no special lexical limitations, that is, with the whole set of possible tags for each word assigned to it when included in the lexicon. With regard to the second suggestion, initial parameters proposed by the Xerox Tagger developers have been preserved in order not to introduce more complexity to the initial parameter estimation. Finally, the choice of the training corpus has consequences on the accuracy of the system. As demonstrated by Merialdo (1994), when using a HMM, a larger training corpus does not necessarily guarantee a better accuracy. On the contrary, an initial model estimated by performing Relative Frequency (RF) training on a tagged text may degrade if a relatively large untagged

corpus is used next. We don't have the possibility of performing a combined (RF and ML) training but, in any case, the potential degradation of the model has been taken into account when producing the final model.

The model has been initially tuned by means of the addition of both transition and symbol biases. For each of the tagsets, three different initial models have been created: one uniform model and two models including biasing facts. The first of these 'biased' models includes the following transition and symbol biases (*Level 1 biases*):

(a) The bigram *clitic – finite verb* is favoured.
(b) The bigram *determiner – nominal* is favoured.
(c) *Conjunctions* and *prepositions* are preferred to *letters of the alphabet.*

The second 'biased' model also includes the following transition and symbol biases (*Level 2 biases*):

(a) Bigrams to force NP structure (especially biased to treat agreement).
(b) Bigram contexts for the *preposition/conjunction* distinction.
(c) When an entry is ambiguous, *nominal* readings are preferred to *adjectival* readings.

Given the relatively large training times needed, every model has been trained using only small training corpora of various sizes. This should not affect the final results since, as it was observed during previous experiments on training with two different tagset sizes, best results were obtained with as few as 50,000 training words. Moreover, the learning curve showed a declination precisely at this point.[5] However, the most promising models have also been trained with larger training corpora in order to observe the whole learning curve.

In order to test every model and tagset, different training corpus sizes have been established. While the whole ITU corpus could have been used as training corpus, the above-mentioned claim in Merialdo (1994) made us take a more conservative line and test out increasingly larger corpus sizes, starting with very small ones.

10.5 Results

Results obtained are presented in Table 10.1. The best score for each model and training corpus size is given in bold. As can be observed, the models using intermediate tagsets produce similar results to those obtained with tagsets 0 and 5, but none of them is markedly better than the rest. With a uniform model, tagset 5 produces the best results, confirming the widespread idea within the NLP community that a small tagset is re-

Table 10.1 Statistics using the 0–5 tagsets

Model	Training files	Word count (with wc)	Errors tagging test corpus 0	1	2	3	4	5	Accuracy 0	1	2	3	4	5
Uniform	No training	0	1609	1612	1668	1829[a]	1634	1480	82.8	82.8	82.2	80.5	82.6	**84.2**
	010–012	19038	1075	1071	1145	1129	1105	997	88.5	88.6	87.8	88.0	88.2	**89.4**
	010–015	34199	1144	1133	1201	1099	952	661	87.8	87.9	87.2	88.3	89.8	**92.9**
	010–018	68628	1116	1111	1191	1065	841	493	88.1	88.1	87.3	88.6	91.0	**94.7**
	010–034	188882	1124	–[b]	1185	1067	832	504	88.0	–	87.4	88.6	91.1	**94.6**
Level 1 biases	No training	0	1548	1558	1614	1660	1556	529	83.5	83.4	82.8	82.3	83.4	**94.4**
	010–012	19038	1080	1055	1130	1267	1199	548	88.5	88.7	87.9	86.6	87.2	**94.2**
	010–015	34199	1089	1099	1166	1140	985	648	88.4	88.3	87.6	87.8	89.5	**93.1**
	010–018	68628	1070	1067	1146	1138	951	1413	88.6	88.6	87.8	87.9	89.9	**84.9**
	010–034	188882	1081	1067	1141	1128	937	1113	88.5	88.6	87.8	88.0	90.0	88.1
Level 2 biases	No training	0	589	591	653	745	696	876	**93.7**	93.7	93.0	92.1	92.6	90.7
	010–012	19038	355	370	438	597	602	793	**96.2**	96.1	95.3	93.6	93.6	91.5
	010–015	34199	403	423	476	550	497	803	**95.7**	95.5	94.9	94.1	94.7	91.4
	010–018	68628	386	377	452	492	436	639	95.9	**96.0**	95.2	94.8	95.3	93.2
	010–034	188882	374	370	442	516	443	483	96.0	**96.1**	95.3	94.5	95.3	94.8
	010–054	334257	375	379	448		445	379	**96.0**	**96.0**	95.2		95.3	**96.0**
	010–079	570225	374	379	449		445	376	**96.0**	**96.0**	95.2		95.3	**96.0**
	010–106	903666	372	377	448		442	390	**96.0**	**96.0**	95.2		95.3	95.8

[a] This model fails to assign the correct tag when observing the ambiguity classes (:alfs :c) (letter of the alphabet-conjunction) and (:alfs :prep) (letter of the alphabet-preposition), assigning always the former. [b] Out of heap

quired when training a HMM without initial probabilities. An increasing learning curve can also be noted with this model and tagset, reaching rates only slightly lower than those obtained with initial probabilities.

However, with the appropriate biases, a model using a linguistically motivated tagset can produce output with similar (or even higher) accuracy. Besides, a more fine-grained tagset allows more subtle biases, thus providing more effective weighted estimates even without training. This fact explains the results obtained with tagset 0 and level 2 biases without any (or a very small) training corpus as compared to the respective models using tagset 5.

Finally, the results obtained confirm the good choice made with respect to a tagset that was both linguistically motivated and capable of outputting at a high accuracy when used in conjunction with a stochastic tagger.

10.6 Language Independence

Two different approaches can be adopted when assessing the language independence of the Xerox Tagger, namely the language independence (i.e. the configurability) of the software package so as to deal with other languages, and the language independence of the probabilistic (HMM) model used by the tagger.

10.6.1 The software package

Although the authors of the Xerox Tagger claim that the whole package is language independent, thus configurable to tag corpora in other languages using the appropriate linguistic resources, this is far from the truth. Apart from 'minor' aspects such as the handling of 8-bit character sets (which are, in principle, supported by the tagger, although the routines for accessing the sorted lexicon have problems with alphabetically sorted files), other limitations have been identified so as to prevent consideration of the whole package as truly language independent. These include (a) tokenization, and (b) handling of linguistic information.

- **Tokenization** Tokenization within the Xerox Tagger has been addressed in the usual way in compilation of programming languages, i.e. specifying token classes with regular expressions and compiling them into a single deterministic finite state automaton. This strategy is enriched with a simple **lookahead** mechanism which allows specification of right context in order to disambiguate certain token classes. This right context is not actually consumed, but only serves the purpose of

disambiguating a given token and hence it is used as the first component of the next token. Lookahead is helpful in identifying sentence boundaries, for instance.

However, the strategy described is clearly insufficient for certain phenomena occurring in natural languages – textwords that span over more than one orthographic word (at least continuous invariant multiword units – see Section 2.2 (1)). Hence, the Xerox Tagger lacks a mechanism for tokenizing fixed phrases or at least 'preparing' them to be appropriately tokenized by the existing tokenizer. Since this is a major issue in certain languages, tagging will be inappropriate for these languages even at the token level.

In the implementation for Spanish, this shortcoming has been solved by means of a pre-processing phase. Space characters separating components of a complex textword are replaced by a tilde (~), so that normal tokenization may operate on these items. This solution is far from satisfactory for certain cases where, depending on the context, a candidate fixed phrase may or may not have a compositional tokenization, but it is better than nothing.

- **Handling of linguistic information** English is a language that poses no specially interesting challenges to (inflectional) morphology. In fact, it is common to attribute the lack of development of theoretical morphology during the 60s and 70s (as compared to theoretical syntax) to this fact, given that most of these developments were focused on English. This tendency has been imported somehow into NLP systems, which have taken syntax, and hence parsing, as the central issue in computational linguistics over a long period of time.

  On the other hand, morphological disambiguation of texts has been usually performed, also for English, by means of programs that, apart from using probabilistic techniques, are based on closed lexicons showing, as in the case of the Xerox Tagger, a relevant tag (or set of tags) assignable to each of the listed wordforms. There is no concept of linguistic analysis in the tagging process but rather only that of selecting the most probable tag.

  This fact is responsible for the lack of morphological components not only in such NLP applications as taggers but also in early parsing systems. The Xerox Tagger is not an exception to this design strategy, and does not make use of a morphological component.

  The situation is aggravated because the lexicon file is a plain ASCII file 'alphabetically' sorted by means of the UNIX command **sort**. This means that a lexicon may need a lot of disk space and, even worse, the same amount of heap since it is loaded in main memory before training

and/or tagging. For Spanish, a plain fullform lexicon derived from about 40,000 lemmas expands into a 13Mb file, with more than 440,000 lines (wordforms). In order to train a model with this lexicon, a prohibitive amount of RAM is needed.

For languages producing more forms per lemma such as, for instance, Basque, the lexicon file may grow to a prohibitive size, making even tagging impossible.

The figures above for Spanish do not take account of complex orthographic words comprising more than one textword – for instance, verbal forms with enclitic pronouns, which are very common in Spanish and in Romance languages in general. In a very conservative approach to these forms, considering the attachment of only two pronouns at most to only four verbal forms (infinitive, gerund and two imperative forms), and granted that clitic pronoun ordering is taken into account, there are ($37 \times 4 =$) 148 more possible forms.[6] In fact, the number is higher. For other languages, the situation may be even worse – in Basque, for instance, verbal forms include a morpheme for every argument in the thematic structure of the verb; Finnish, Hungarian and even Basque show a very rich system of positional cases. Consequently, and solely because of this design decision, it is difficult to consider the Xerox Tagger a language independent system.

The port to Spanish preserves the structure of the lexicon file, and still lacks a morphological processor,[7] but includes a module handling verbal forms with enclitic pronouns, so every separate word identified in these multiwords is assigned one tag.

Finally, the way ambiguity classes for unknown words are approximated is considered inadequate (because of its inefficiency) for Spanish, although, in a sense, it would work for any language.

10.6.2 The probabilistic model

Probabilistic models can be built for every language. However, the relevance and, thus, the accuracy of these models may differ from one language to another depending on two axes of the language considered – that of lexical ambiguity and that of context ambiguity. Let us call **lexical ambiguity** the average number of tags per wordform and **context ambiguity** the average number of tags that may precede a given tag (since we are using a bigram model). We can now assume that languages showing a higher lexical ambiguity are less contextually ambiguous and, conversely, languages with a lower lexical ambiguity are potentially more contextually ambiguous. This seems to be the case, since languages

showing a higher number of homographs have also stricter word ordering, while languages with more wordforms (and hence fewer homographs) show a relatively free word ordering.[8]

In this scenario, a simplification of the probabilistic model along the context axis, such as that carried out by Cutting and Pedersen, may pose no major consequences to languages with a high lexical ambiguity but with a low context ambiguity, as is the case for English. On the other hand, for less lexically ambiguous, but, because of that, more contextually ambiguous languages, like Spanish, this simplification brings a reduction in the overall accuracy.

In this respect, the results obtained with the ITU corpus (characterized by its very specialized technical domain, giving rise to restricted use of language in the lexical, morphosyntactic – only third person verbs – and syntactic levels) must be considered with care. Tagging experiments with other less restricted corpora have shown an unsurprising decrease in tagging accuracy of 2–4 per cent.[9] This decrease would have been certainly palliated with a trigram model. Nevertheless, what remains to be investigated is the behaviour (maybe the limitations?) of the trigram model with languages showing free word order.

10.7 Conclusions

This chapter has dealt with the issue of retargeting a well known public domain tagger originally implemented for English – the Xerox Tagger. With some modifications, necessary in our view, for tagging inflectional languages and for the proper segmentation of complex words, the system behaves with the usual error rates accepted for other morphosyntactic taggers, although a more subtle view of the results obtained leads us to think that the use of a first-order model is responsible for the slightly lower accuracy as compared to results for English.

It has also been proved that a fine-grained tagset does not necessarily carry more ambiguity and can, consequently, be used with a probabilistic tagger with reasonable results. This brings the benefit, for the tagged corpus, of including the whole variety of morphosyntactic categories and subcategories which are important for Spanish.

Notes

1. This project was funded by the Commission of the European Union under the contract number MLAP-93/20, and ran from February 1994 through September 1995.

2. These tags represent, respectively, a Feminine Singular ADJective, a Feminine Singular Common Noun, a Third person Singular Present Indicative Lexical Verb, a First person Singular Present Subjunctive Lexical Verb and a Third person Singular Present Subjunctive Lexical Verb. Verbal forms belong to verbs from different conjugations, thus there is no single verb showing forms ending in *a* with these tags.
3. See Sánchez León and Nieto (1995) for a demonstration of the benefits of this model. Recent proprietary versions of the Xerox Tagger (Chanod and Tapanainen 1995, Feldweg 1995) also include guessers using knowledge-based rather than statistically-computed morpheme information.
4. The ambiguity rate has been measured in the lexicon rather than in the corpus because (a) the lexicon contains all and only the words in the corpus, and (b) it contains all possible tags for every word and not only those found in the corpus. The only exceptions to this are first and second person singular and plural for verbs, which, not occurring or marginally occurring in the ITU corpus, have been removed from the lexicon. The ambiguity rate is measured over all the lexicon entries and not just over the ambiguous ones.
5. With 30,000 training words, the tagger showed a 95.68 per cent accuracy; with 50,000, the accuracy was 95.93 per cent; finally, with about 70,000 words, it decreased to 95.29 per cent.
6. Assuming that every verb may have two objects or one object plus a *se* pronoun. Nevertheless, in order to filter out all these possibilities and leave only the truly possible ones, the lexicon should include valency information for verbs.
7. There is a burden in plugging in such a component since, to calculate the complete set of ambiguity classes, the whole fullform lexicon must be created at least once. In any case, the time constraints imposed by needing to have a working tagger in a relatively short period of time prevented us from including such a morphological component.
8. But note, incidentally, that the ambiguity rate in the lexicon derived from the Brown Corpus (included in the tagger package) is only 1.1068 tags per word, although it only contains tags actually occurring in the corpus and no other (possible) tags for every word.
9. This, however, contrasts with results obtained in ports to other languages, that have used a relatively balanced corpus. Thus, Chanod and Tapanainen (1995) report 96.8 per cent accuracy, while the accuracy reported for German is 96.66 per cent (Feldweg 1995).

11

The Use of Syntactic Annotation Tools: Partial and Full Parsing

JEREMY BATEMAN, JEAN FORREST and TIM WILLIS

Chapter 3 presented the topic of syntactic annotation of corpora, and briefly exemplified a wide range of different parsing schemes. In this chapter, we turn to the methods of annotating a corpus – or building a **treebank** (see Section 3.2). Our major concern here is not with the software itself, but with the way software and the human analyst interact in treebank production. Both human and machine have an essential role to play. In the latter part of this chapter, we focus on a particular innovative model of treebank production, which has been developed in a collaboration between ATR (Kyoto) and UCREL, Lancaster.

11.1 Methods of Syntactic Annotation

One of the major aims of NLP over the past ten years has been to produce a **wide-coverage** 'grammatical analyser'[1] or **parser**. For many NLP applications, the challenge is to produce a parser which will automatically be able to structurally analyse correctly, according to a defined parsing scheme, any sentence of naturally occurring **unrestricted** English, from as wide a range of genres as possible. How can this robust parsing be achieved? Currently, the consensus, as with grammatical tagging (see Section 10.1), is in favour of a probabilistic approach.[2] However, other methods, which do not use statistics, are also achieving significant success.[3]

An English-language parser that works is one that correctly analyses almost any randomly selected English sentence, correctly demarcating it into its labelled constituents and labelling each word with an appropriate part-of-speech tag. In order to train and test a statistically developed grammar/parser, it is necessary to have a large database (treebank) of sentences annotated to show their structural features. To achieve this requires the annotation (by human processing, machine processing, or a combination of both) of a large quantity of text, parsed according to a

defined scheme. At present, there is no automatic parser which can produce an accurate enough result for full and satisfactory treebank development,[4] so human analysts normally have to be involved in the process.[5] This work requires the development of tools to help the analyst.

In practice, however, the division of labour between human and machine may vary a great deal. At one end of the scale, the human analysis is predominant, at the other end automatic analysis is predominant, whereas in the middle a more interactive model prevails. In fact, we may distinguish the following as positions along the scale of human-to-automatic processing:[6]

- Human analysis and input (Ellegård 1978, Sampson 1987b)
- Human analysis enhanced by purpose-built editor (Garside, EPICS)
- Automatic parsing interactive with human editor (TOSCA)
- Automatic parsing with correction/completion by human post-editors (Penn Treebank)
- Automatic (partial) parsing, with manual post-processing omitted (ENGCG, Bank of English)

What is corpus annotation aiming to achieve? Firstly, it is necessary for the annotation to be based on a clearly defined **parsing scheme** (see Section 3.2), in terms of constituents and labelling, so that the manually parsed treebank can be used as a testbed for grammar development and training, and also as a resource for linguistic research. Other criteria include (1) a consensual, rather than theory-based approach, and (2) as wide a language coverage as possible so that all the subtle nuances encapsulated in real-life data, as opposed to the artificiality of grammarians' hitherto theory-driven examples, can be exemplified in the database.

One of the aims of corpus annotation is to capture idiosyncrasies of ordinary language which apparently have not been addressed before, whilst not opening up loopholes in the grammar which might allow nonsensical parses to proliferate. For example, from a weather forecast in the ATR treebank (Black *et al.* 1996) we have 'Wind EAST 10 to 15 MPH'. The following is not necessarily an ideal or perfect parse, and other solutions could have been found, but this is how, in outline, one Lancaster analyst parsed this sentence, using an existing adverbial rule (r) to postmodify the head noun (n), making it into a postmodified noun phrase (N'):

[S [N' [n WIND_NN1 n] [i [r EAST_RL [n [d [m 10_MC [p TO_IITO [n [d 15_MC d] n] p] m] d] [n MPH_NNU n] n] r] i] N'] ._. S][7]

By comparison, Sampson (1995: 277) parsed the rather similar telegraphese style construction *Nuclear cloud over Siberia after blast* by the use of an L (verbless clause) constituent:

[Oh [L [Ns Nuclear cloud] [P over Siberia] [P after blast] L] Oh]

How can the need for productivity be balanced with the need for accuracy? Obviously, the latter can jeopardize the former requirement, but quality and quantity must both be achieved. Analysts must aim for consistency within their own work, consensus and consistency with other analysts, and linguistic validity. The tighter and more complex the scheme of analysis, the longer the learning curve and thus the slower the initial rate of productivity.

11.2 The Approaches Taken by Different Projects

Different projects have tackled these problems on an *ad hoc* basis, learning and deriving some insights from shared experience of other projects, but also developing their own solutions. To some extent, the way parsing projects have evolved is dependent on variations in what they are aiming to achieve. For example, Sampson[8] cites his own experience with SUSANNE (see Section 3.3.4) and his own criteria, in comparison with those of the Penn Treebank (see Sections 3.3.1–3.3.2), but accepts both ways of achieving goals as equally valid (the Penn project used the 'automatic parse with manual correction' methodology):

> The SUSANNE Corpus was produced as an adjunct to the development of detailed analytic standards; consequently it could only be as big as was compatible with individual attention (often, attention by several individuals) to almost every difficult analytic decision posed by its language ... The motive of the Penn treebank ... is to produce the largest possible quantity of analysed language material, using an analytic scheme which is only as subtle as is compatible with that aim ... Both of these alternative research strategies are as valid in their own terms.

The aims of the SUSANNE scheme were to be comprehensive, explicit and authoritative whilst the Penn Treebank project, at least in its first phase, seems to have taken a more pragmatic approach, in which quantity was more important. The Penn project, led by Mitch Marcus, has produced treebanks of at least 2.5 million words, using the FIDDITCH parser, whose skeletal parse output was edited by treebankers, making decisions left unmade by the parser (see Marcus *et al.* 1993).[9]

One of the earliest examples of manual parsing was Sampson's pioneering effort at Lancaster, in the mid 1980s, when he parsed over 40,000 words of the LOB Corpus, subsequently known as 'the Lancaster-Leeds treebank' (Sampson 1995: 16–19). This was in response to the need of the automatic parser for a manually-analysed database as a source of statistical information, stemming from work on statistics-based parsing

techniques led by Leech (who claims the term 'treebank' originated around this time) and Garside in the early 1980s (Garside and Leech 1985). This was also the genesis of the SUSANNE scheme (Sampson 1995), for which Sampson also drew on the even earlier achievement of Alvar Ellegård (1978), who (with his students) produced a larger database of parsed texts but using a less rigorously defined parsing scheme. In these early efforts, the human analysis was predominant, and little use was made of automatic means for input, editing, etc.

A more interactive model has been developed since the early 1980s at Nijmegen University, where the TOSCA research group (see Section 3.3.3) concentrates on creating an integrated environment for the compilation, storage and analysis of annotated corpora, along with corpus compilation and exploitation. TOSCA also pays attention to grammar development, treebank editing and selection of contextually appropriate parses. The role of the computer is restricted to first-choice options, backed up by alternatives; that of the linguist is to provide semantic, pragmatic and extra-linguistic information. TOSCA parsers are based on separately-developed computer grammars. In practice, the TOSCA grammars are gradually constructed in a cyclical interaction between the linguist's knowledge and grammar testing procedures.[10]

The recently founded Research Unit for Multilingual Language Technology (MLT) at Helsinki (led by Karlsson, Koskenniemi and Voutilainen) lists some basic principles underlying past research which will determine future research: 'The theories and methods which are developed are intended to be language independent ... The grammars written should cover ordinary unrestricted running text ...' and there should be collaboration with other research centres and scholars. The Helsinki team's goals include 'further developments of surface-syntactic parsing methods', and 'techniques for semi-automatic discovery methods assisting the creation of grammar rules'. Their achievements up to now include a general Constraint Grammar parser (see Section 3.3.5), making possible full-scale surface syntactic analysis at a speed of 600 words per second, given optimum hardware.[11] The ENGCG method is one of partial dependency parsing: perhaps this is the nearest thing to totally automatic treebank creation, although the output of ENGCG is not a treebank in the accepted sense.[12]

11.3 Problems of Grammar Size and Power

Anyone who tries to build a treebank consisting of every sentence encountered in a varied corpus soon realizes that the grammar of a language such as English is immense: so immense, that the ideal of a 'complete grammar of English' can probably never be attained. This means that a treebanking

project relying on human analysts has a problem of control. On the one hand, as more text data are analysed, the grammar needed to account for them gets bigger and bigger. On the other hand, the grammar has to be powerful enough to limit the number of ambiguities per sentence. For long and complex sentences, the number of ambiguities permitted by a grammar may run into thousands or even millions. This problem gets worse as a grammar itself becomes larger and more complex.

Before going on to discuss in some depth Lancaster's current collaboration with ATR Japan, using a full grammar and the Grammarian's Workbench Tool (Black *et al.* 1996), we will briefly touch on Lancaster's own experience (see also Leech and Garside 1991), through collaboration with IBM, of phases of treebank development, starting in 1987 with the development of a grammar by hand. Bearing in mind the criteria necessary for producing a treebank of manually-parsed sentences on which a statistical grammar could be trained and tested – size, speed, uniformity and accuracy – a 'task force' of grammarians was recruited to manually parse one million words, after a suitable and thorough training period, using a feature-enriched phrase structure grammar which each parse was checked against and had to conform to. An inputting program was available for analysts to input parses laboriously worked out on computer printouts. However, the rules expanded rapidly, as new structures were encountered in almost every sentence, and the cumbersome procedures of approving rule changes, updating the grammar, and checking and validating the parses against the grammar meant that the work was progressing too slowly and with too much inconsistency to be of use. An example of a parsed sentence is given below:

```
[S'[B[R Here_RL R] ,_, B][Sd[N'[Da the_AT Da][N government_NN N]N']
[V'[V guarantees_VVZ [Fn that_CST [B[Fa if_CS [Sd[N' you_PPY N'][V'[V
put_VV0 [N'[Da your_APP$ Da][N money_NN1 N]N'][P in_II [N'[Da a_AT1
Da][N savings_NN2 account_NN1 N]N']P]V]V']Sd]Fa] ,_, B][Sd[N' it_PPH1
N'][V'[Od will_VM n't_XX Od][V get_VV0 [Vn[R totally_RR R] eaten_VVN
[R up_RP R][P by_II [N'[N inflation_NN1 N]N']P]Vn]V]V']Sd]Fn]V]V']Sd]
```

Clearly this model of treebank development was not a successful one, and had to be replaced by a new one. The method adopted in 1988 was to uncouple grammar development from treebank production (see Garside and McEnery 1993). At this time, for the treebank development, a simple scheme of sentence analysis known as **skeleton parsing** (see Section 3.2, Eyes and Leech 1993) was adopted, and led to an almost immediate escalation in productivity. In practice, more detailed guidelines evolved, quality control was introduced, and analysts moved more closely towards consensus and consistency among themselves.

Equally important in providing a comfortable and optimum environ-

ment for the task of inputting the bracketing of the skeleton parse were the corpus annotation tools, in particular the CLAWS automatic tagger, the LB editor for post-editing the output of CLAWS, and most importantly Roger Garside's special purpose editor for the assignment of labelled bracketing – EPICS (Garside and McEnery 1993). This is an editor for inputting annotations manually, but an experienced and trained analyst can annotate a sentence in less than a minute, which includes reading, assimilating the structure and inputting the parse. When the collaboration with IBM was concluded in 1992, about 3 million words of text had been skeleton parsed.

The uncoupling of treebank and grammar development meant that the grammar development could follow a different path, while maintaining basic consistency with the treebank. The grammar adopted (Black 1993a) was a more powerful model described by Black (*ibid*: 144) as a 'feature-based context-free grammar employing a weak version of unification'.

11.4 A Human/Machine Treebanking Environment: Lancaster/ATR

This section will deal with the work UCREL has been undertaking for Advanced Telecommunications Research (ATR) of Kyoto, Japan. The collaboration between ATR and UCREL began in August 1994, and has involved the syntactic and semantic tagging, and parsing according to the ATR English Grammar (developed by Ezra Black), of more than a million words of spoken and written English from a variety of sources.

The ATR grammar is a comprehensive general coverage grammar of English. Analysts initially trained themselves on this grammar using text data from computer manuals, for which the grammar made particular provision, e.g. in the form of noun phrase rules which have been found to recur in such data. Subsequent work with general English has entailed the continuous addition of new rules to cope with new structures, for example verb phrases with more or different complements than were provided for in the original grammar.

In terms of the division of labour between human and machine, this is a fully interactive system: the parse is interactively created using a complex human-computer interface. The separation of treebank development and grammar development, which was a feature of the Lancaster/IBM collaboration, no longer applies here: a powerful grammatical analysis tool (**GWB**, or **Grammarians' Workbench**) is able to 'collaborate with' the analyst in offering only parses which are conformant to the currently stored version of the grammar. The overall design of the

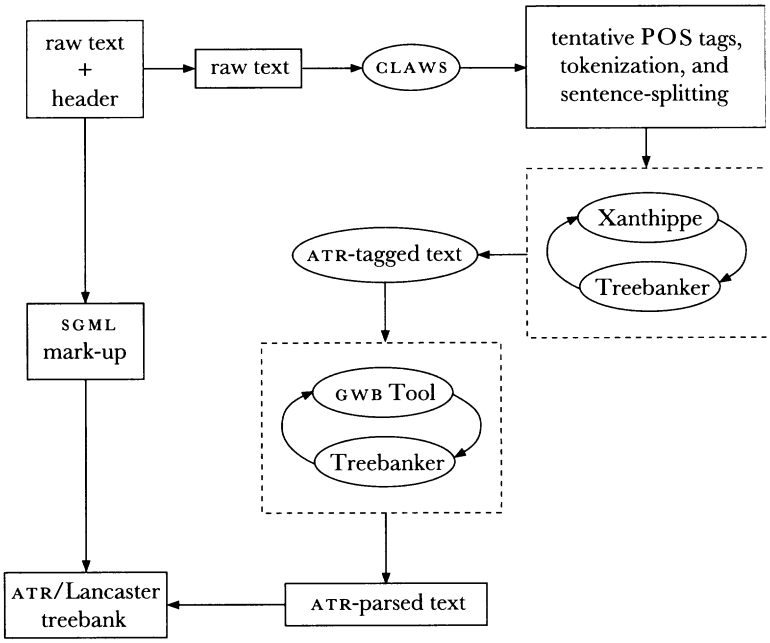


Figure 11.1 The annotation process for the ATR/Lancaster treebank. Portions inside dashed boxes represent interaction between the human treebanker and the software tools

treebank development environment, including both human and machine processing phases, is shown in Figure 11.1.[13]

11.4.1 Training and interaction between the analysts and grammarian

The initial contract between ATR and Lancaster stipulated a training period to familiarize the team of analysts with the tagset, the grammar and the GWB tool. The initial analysts had already accumulated considerable expertise in syntactic tagging and skeleton parsing, but found themselves having to 'unlearn' practices learned on other schemes which were inappropriate to the ATR grammar (see below). Knowledge of skeleton parsing helped only the early stages of analysis of each sentence, as analysts had to continually check analyses which might produce parse trees of similar shapes but which utilized subtly different rules.

The training period involved using the Xanthippe tag editor (see Section 13.3.1), designed at Lancaster, and the GWB tool, provided by ATR.

GWB provides an interface between the analyst and the ATR grammar, and will analyse any sentence whose every text item (including all punctuation) has been assigned a valid tag. All parses are displayable in a window which the tool generates. Nodes indicate the rule governing the constituent underneath. Coloured nodes contain the label of the rule governing the subordinate constituent; clicking the node with the left mouse button displays a pop-up screen containing the features associated with this rule in this situation, such as number, type of noun, and head semantic content. White nodes indicate that other options are available for the constituent, and may offer a number of further parses for many constituents.

In order to ensure optimal performance, the Lancaster analysts (Bateman, Forrest and Willis) engage in constant interaction by email with the grammarian (Black). Other forms of communication were attempted early in the project, including regular telephone calls and Talk, a system in which the two parties to a conversation can send individual lines instantaneously and get answers as they type. Telephone proved insufficient as it sacrificed the facility of working out both questions and answers before answering; it was also more costly than electronic communication. Talk proved difficult to express complex grammatical ideas through, and to coordinate. Both parties resorted to the more familiar medium of email to explore the questions raised in sufficient depth.

Boundaries between semantic tag definitions have produced a continuing source of discussion as analysts have approached data from new subject areas. In the early stages, work began on the English of computer manuals. In moving from this to work on General English, the analysts interacted with the grammarian to suggest new categories which might be needed. This led for example, to a diversification in the tagging of 'concrete' nouns, of which those in the computer manuals domain had all justified the tags NN*DEVICE and NN*DEVICE-PT (where PT indicates 'part'). To accommodate widening requirements in the corpus, the analysts suggested adding the tag NN*SUBSTANCE, for solid objects which were not artificial, and NN*FOOD.

Analysts have also asked detailed questions as to distinctions between various very similar grammar rules. These questions can lead to in-depth discussion of the grammar's approach to certain problems. Often the grammar allows different parses for constructions which are open to slightly different interpretations, and the grammarian has advised on which parse to select here (see below).

Typically analysts have sent queries at the end of each day, covering problems which have arisen during that period. Problems may be solved by reference to existing aspects of the grammar or tagset, or else the grammar will need to be altered. A new grammar is sent from Kyoto, via email

and FTP, each week and loaded into the tool. This will include rule alterations, allowing constituents or semantic tags in places where they had been denied before, or adding new rules as more complex constructions have necessitated them (the number of verb rules, for example, has grown a fair amount during the project). However, new rules are needed to cover infrequent constructions only, about 90 per cent of the textual material being parsed without any need for updates.

Due to the emphasis on maintaining a high volume of parsed text, analysts will typically leave a portion of each text unparsed in an associated query file. These sentences will sometimes contain many most interesting constructions, but are left for the grammarian to approach when the file is completed. As new text types arise, their stylistic idiosyncrasies sometimes necessitate new rules, and analysts ask for guidance as it becomes apparent what the grammar requires in order to cover these new constructions.

11.4.2 Part-of-speech tagging

POS (word-class) tagging was the first, and simplest, aspect of the work to master. The tagset of the ATR Grammar, as far as its strictly syntactic aspect is concerned, is fairly closely related to the CLAWS tagsets, and so a first step is to run the tagger CLAWS over the data (see Chapter 7). Analysts then use the Xanthippe tag editor (see Section 13.3.1) to correct automatic tag selections when necessary and select semantic tags, which are an essential element of the ATR parsing but which have not been addressed by previous parsing schemes implemented at Lancaster. Differences between the CLAWS and ATR morphosyntactic (POS) tagsets caused inconsistencies between data in the corpus produced earlier and later, so the grammarian has responded to consistent mistagging by reminding analysts of correct tags. For example, only late in the project did analysts use VMPAST consistently for *could*, *would*, *should*, *might*, and particularly the enclitic *'d*, which CLAWS consistently tagged VMPRES throughout the first (written language) phase of the project and which analysts have only consistently amended to the required VMPAST in the second (spoken language) phase. The one grammarian receiving output in Kyoto has a considerable amount of work to do in checking its correctness and consistency, and no parse is affected by incorrect selection of modal tags, so this (the only case of such persistent mistagging) did not stand out enough to be the subject of major corrective effort until spoken data produced a much higher frequency of *'d* in more recent work.

For content words Xanthippe offers an interface to the relevant ATR semantic tagset.[14] When CLAWS, or an analyst, selects a syntactic tag

for one of these categories, the default word *semantic* is displayed. This word is a button, and when it is clicked a window appears offering all the semantic tags appropriate to that word class. Analysts click the tag they consider most correct, which is then displayed.

Appropriate use of the semantic tagset has been the result of evolution rather than of prescription. Analysts have decided which tags they felt appropriate based on these guidelines and their own knowledge of the real world represented in texts. Analysts have mostly worked in the same room, so they have been able to compare selection criteria among themselves throughout the project. However, some ambiguity has necessarily remained.

The grammarian has emphasized that tags for individual words must be decided on the basis of individual cases. For example, in a document about alleged sightings of UFOs by astronauts, *control* was tagged FUNCTION and SYSTEM-PT as a common noun, and ORG as a proper noun:[15]

Do you still have control_NN1FUNCTION
They received another message from control_NN1SYSTEM-PT.
Control_NP1ORG, this is Gemini 7

Further tags to be considered for *control* are INSTIT (for the proper noun) and INTER-ACT (for the common noun).

11.4.3 Use of a full parsing tool: the GWB tool

To work effectively, analysts' knowledge of the grammar has to be matched by their knowledge of how the tool presents parse options it produces. As they grew familiar with the grammar, analysts learned which grammar rules the tool would prioritize from those permissible for certain constructions. Analysts would, for example, come to recognize that a simple sentence might generate a noncontroversial parse tree, but there may be two verb rule options available, one of which indicates the presence of a 'missing' element in the structure. The grammar assumes that such missing elements can occur in a wide variety of positions, thus multiplying the number of parses for many sentences by offering trees which suggest the presence of these elements as well as those which do not.

Any parse which fits the grammar will be given as a potential parse. Most choices of parse involve consideration of common constructions and analysts will recognize which parse is appropriate immediately. In the unusual cases where analysts are unsure which parse to select (e.g. when two appear precisely the same but for minor discrepancies in rule labelling), they either email to request clarification or enter the problem in the query file. The grammarian discusses most matters raised in emails,

either recommending which parse to choose, or deferring judgement, in some cases because the structure is too rare to be worth incorporating into the grammar immediately, although at a later time it would need to be included.

11.4.4 Bracketing in ATR parsing

In a manner developed from the skeleton parsing described above, analysts constrain the number of parses the grammar would deem permissible for each sentence of any length by inserting brackets into the sentence to be parsed, demarcating major constituents such as noun and verb phrases, grouped adjectives, determiner phrases, and co-ordination. This constrains the grammar to produce only parses which conform to this basic structure. However, if only major constituents are marked, this may still leave the grammar with a massive range of consistent but utterly wrong groupings of constituents. Take a classic example of the perennial problem of prepositional phrase attachment:

I saw the man in the park with the telescope.

This can be treated as containing a verb followed by a single complement, with *the man* postmodified twice:

1. I [saw [[[the man][in [the park]]] [with [the telescope]]]]

Or we can label *the telescope* the instrument of the action, the verb phrase consisting of verb, noun phrase, and one further complement:

2. I [saw [[the man] [in [the park]]] [with [the telescope]]]

Or we can separate *in the park* also, making a more complex verb phrase with two prepositional phrases as complements:

3. I [saw [the man] [in the park] [with the telescope]]

In practice, analysts have developed ergonomic expedients to permit them to arrive quickly at a small set of possible parses.

11.5 Practicalities of Large-scale Machine Parsing

The complexity of language is such that a grammar which purports to parse a comprehensive range of English sentences seems to have the drawback that it will generate a huge selection of invalid parses. At present, it seems essential to use human analysts to select the 'correct' parse or parses. A parser which uses a closely defined grammar can still present the

analyst with the problem of choosing a single 'correct' parse, if such is required by the project, when several are available which appear equally valid but none of which covers every nuance captured by the collection of 'correct' parses available. This can lead to the presence in the treebank of a variety of equally correct solutions (cf. Section 3.2) for a given construction, as it occurs in different sentences. Corpus projects involving human grammatical analysis, for their part, can strive for consistency through thorough training of analysts in agreed approaches to particular problems and effective collation of decisions on ambiguities of analysis, available to the co-ordinating grammarian(s) as well as the analysts themselves. This chapter has presented the point of view that selection of 'correct' parses by human interpretation in context is the only sure road to the progressive improvement of automatic parsing of unconstrained text.

Notes

1. See Black (1993a) for a overview of different corpus parsing models, statistical and non-statistical.
2. On statistical/probabilistic/stochastic parsing, see especially Jelinek and Lafferty (1991), Black *et al.* (1992), Black *et al.* (1993), Briscoe and Carroll (1993), Stolcke (1995).
3. Examples of non-probabilistic approaches to corpus parsing are those of Hindle's FIDDITCH parser, used for the Penn Treebank; Brill's parser (Brill 1996), and the ENGCG parser of Karlsson (1994), Karlsson *et al.* (1995).
4. See the evaluation of different corpus parsers by Black (1993a). The success rates he reports vary from 78 per cent to 29 per cent. As he points out, however, the ways of measuring success vary considerably.
5. In fact, it will be argued by many that the correct and appropriate parsing of a sentence can be determined only by the human mind, which is able to take account of the most likely meaning(s) of a sentence in its context.
6. These have been introduced in Sections 3.3.1–3.3.5, and will be further discussed in Section 11.2.
7. Here the non-terminals are those of an early IBM/Lancaster treebank.
8. Quoted from Sampson, G., 'SUSANNE and Penn Corpora', http://www.cogs.susx.ac.uk/users/geoffs/RSue.html, 1996.
9. Both the SUSANNE treebank and the Penn Treebank are publicly available. For contact addresses see Appendix I.
10. See van Halteren and Oostdijk (1993); also the TOSCA Research Group www pages, http://www.lands.let.kun.nl/research/tosca/togen.html.
11. See WWW, Research Unit for Multilingual Language Technology, http://www.ling.helsinki.fi/research/rumlat.html.
12. The focus of the Helsinki group has been on parser development rather than treebank (annotated corpus) development. The application of the Helsinki ENGCG to a very large corpus (the Bank of English) is significant in this

respect, in that the view of John Sinclair, originator of the Bank of English, is that no human postediting is necessary or desirable. Sinclair (1992: 381) argues that 'analysis should be restricted to what the machine can do without human checking or intervention'.

13. We are grateful to ATR for permission to reproduce this diagram, which first appeared in Eubank, Kashioka, and Black *A new approach to treebank creation*, in *Proceedings of the Second Annual Meeting of NLP* (1996), pp. 265–8.
14. Different choices are available for nouns, verbs, adjectives, adverbs, and proper nouns, as certain semantic categories are deemed inappropriate for certain parts of speech; for example, the tagset recognizes VERITY as a tag for adjectives and adverbs, and the semantically-related VERIFY for verbs, but there is no equivalent tag for nouns.
15. Here we use the semantic 'suffix' of the tag independently of the morpho-syntactic prefix, which is (for example) NN1 for singular common nouns and NP1 for singular proper nouns. FUNCTION is used for common nouns which are types or names of functions, e.g. in computational or organizational contexts; INTER-ACT for common nouns referring to actions between people, or between people and institutions; SYSTEM-PT for sections, attributes, agencies, etc. of systems; and ORG is a proper noun tag for human organizations.

12

Higher-Level Annotation Tools

ROGER GARSIDE and PAUL RAYSON

12.1 Introduction

In Chapters 7–11, our survey of annotation tools has so far largely restricted itself to the two levels of grammatical tagging and (partial) parsing. Our task in this chapter is to examine briefly the kinds of tools that may be used to annotate a corpus at higher, or more abstract, levels than these: for example, semantic and discourse annotation. The types of annotation that one may wish to make at such higher levels have already been discussed in Chapters 4–5. As the need for these kinds of annotation has only recently begun to make itself felt, the tools for such tasks are in a relatively early stage of development, where they exist at all. However, we can observe two general methods, comparable to those observed for syntactic annotation: (a) interactive hand annotation, using a more or less specialised **editor**; and (b) automatic annotation, the annotation being done (for example) by a **semantic tagger**, with the possibility of hand correction or disambiguation. In practice, to illustrate the kinds of development now taking place, we will confine our attention here to one tool of each kind: an interactive editor for discourse annotation (Section 12.2) and an automatic semantic tagger (Section 12.3).

12.2 A Task-Oriented Editor for Discourse Annotation: XANADU

We have already seen that annotation is often a difficult and time-consuming process if it is to be accurate and cover a substantial amount of data. The annotation process needs to be an appropriate division of labour between manual and machine processing. In our research at UCREL we have been concerned with the optimal interaction between manual

skills and automatic processing, and have developed a series of 'intelligent editors' to aid in the annotation of texts. One of these is XANADU,[1] which enables analysts to annotate text with discoursal information about anaphoric and other kinds of cohesive relations. Section 12.2.1 describes the basic procedures of marking the texts, and the XANADU editor. Section 12.2.2 describes some of the special features of XANADU, and the changes made in the light of analysts' experience in performing the required tasks. The process of the tool's progressive enhancement through feedback from its users is seen as an important aspect of its development. We have plans for the further modification of the design of XANADU for use in other text annotation processes such as prosodic or stylistic annotation (Sections 6.1 and 6.3). Thus, in discussing XANADU, we aim to demonstrate the design principles of a potentially generic editing tool, that may be adapted to various annotation purposes.

12.2.1 The annotation process

As input to the XANADU annotation process, we use texts which have already undergone syntactic annotation in the form of skeleton parsing (see Section 3.2), since it is widely recognized that parsing is an important preliminary to such discoursal tasks as anaphora resolution. Much of the cohesion annotation has been done on a treebank of AP news stories,[2] which had first of all been divided into units of approximately 100 sentences for the syntactic annotation. We have retained this block size as the basic unit for anaphoric annotation. When the AP blocks were selected they always consisted of an integral number of news stories, so there were no anaphoric references across block boundaries. However, a block often consists of a number of news stories, and here the analyst makes use of the cohesion barrier symbol mentioned in Section 5.3.12 (1), to separate distinct cohesive passages of text, between which anaphoric links are barred.

XANADU is an interactive tool allowing an analyst rapidly to mark corpus texts with cohesion annotations, for which we have used the UCREL/IBM discourse annotation scheme described in some detail in Chapter 5. This program is written in C, and runs under X-Windows and UNIX on a Sun workstation. As mentioned above, XANADU has been through various versions, each of which has been used by the analysts to annotate a number of texts, as a basis for further improvement. To make the human analyst's work as simple and efficient as possible, the principal design aim has been to ensure that the simpler, more common types of annotation require the minimum number of user actions, if necessary at the expense of more user actions for the more complicated but rarer cases. For user-friendliness, another design principle has been that most of the annotation

be performed with the mouse, rather than by typing information at the keyboard.

An illustration of the situation when an analyst is the process of annotating a text is given in Figure 12.1 (overleaf). The screen is divided (disregarding the pop-up menu shown at bottom left) into three main areas.

(a) *Text area* At the top of the display is a portion of the text to be marked, in a window with a scroll-bar to allow the analyst to move through the text. Although the text-file contains part-of-speech markings on each word, and syntactic brackets round significant constituents, these are not displayed in the window, as they would clutter the text and make it difficult to read for sense. It is, however, possible to look at the syntactic marking of a particular part of the text if desired (see Section 12.2.2).

(b) *Action area* At the bottom left-hand corner of the display are three sets of 'action' buttons. The main set, at the top, is for inserting the various cohesive markings, with one button for each general type of marking – for example **anaphoric/cataphoric** (proform) reference, **substitute form**, **ellipsis**, **cohesion barrier**, etc. (see Section 5.3). Below these is a set of editing buttons; and below these again is a set of miscellaneous buttons. Clicking with the mouse on one of the annotation-insertion buttons brings up a pop-up window containing a set of further buttons, providing the appropriate options for each particular marking.

(c) *Antecedent area* At the bottom right-hand corner of the display is a list of all the antecedents so far marked in this text, together with the index numbers allocated to them by the program. In the case of a set of non-proform coreferential items with the same index number (e.g. a repeated proper noun), the textually most recent item is the one displayed against the number.

Consider first the marking of a new antecedent, which should have a unique index number associated with it. The procedure is to click the mouse on the beginning and end of the stretch of text to be marked as the antecedent, and then to click on the 'new' button. This brings up a pop-up menu of options for an antecedent, as shown in the bottom left of Figure 12.1, allowing such things as uncertainty about the boundaries of the antecedent to be indicated, if desired.

Finally the 'confirm' button is clicked – this causes an unused index number to be allocated by the program, and the appropriate marking to be inserted in the text according to the options selected. The marking is displayed at the appropriate place in the text window, and the new antecedent is inserted in the list in the lower right-hand part of the display.[3]

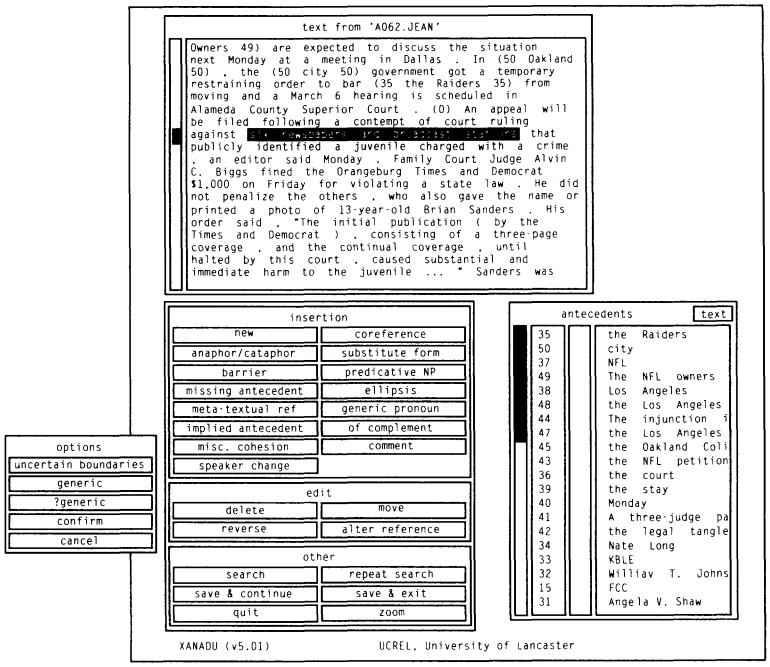


Figure 12.1

Now consider marking a proform (e.g. a pronoun), where we wish to indicate its linkage with a previously-marked antecedent. Here the analyst clicks the mouse on the proform, then clicks on the 'anaphor/cataphor' button. A single click in the text window indicates that the single word indicated is to be marked. The options menu for the 'anaphor/cataphor' button is now displayed (see Figure 12.2).

In a typical simple proform annotation, the only further piece of information required is the antecedent number, which is obtained by clicking on the appropriate line in the list of antecedents (after scrolling the antecedents window if necessary; but if the analyst is working through the text in the normal way, the antecedent will usually be visible in this window, somewhere near the top). The clicking on the appropriate entry in the antecedent list is taken to confirm this marking, eliminating an extra button click in this simple case.

There is provision for indicating multiple or alternative references to two or more antecedents (see Sections 5.3.2 and 5.3.4). Also sometimes the analyst may wish to mark one or more of the reference numbers as uncertain. Both these features are handled with the buttons on the pop-up

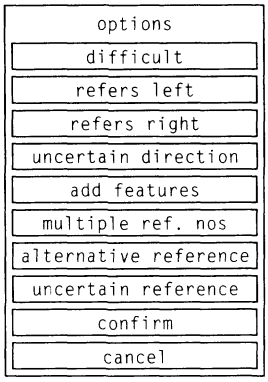


Figure 12.2

list. To indicate multiple references the analyst clicks on the 'multiple reference' button, and then clicks on any number of antecedents in the antecedent window, ending by explicitly pressing the 'confirm' button. (In order to keep track of what references have been indicated, a small window appears showing the reference numbers selected so far.) Variations on this procedure apply to other rare types of annotation such as 'uncertain reference' and 'alternative reference'.

Notice that this mechanism requires the antecedent to be already in the antecedent list. In the case of a cataphoric reference, where the proform appears textually before the antecedent, the normal method of working through the text from beginning to end has to be temporarily abandoned, as the antecedent must be marked before the proform can be marked. As shown in Sections 5.3.2–5.3.4, the annotation for a proform includes an indication of direction towards the antecedent, through the choice between the characters '>' and '<'. The program makes an automatic attempt to choose the correct direction; but in the rare case of multiple references, where the reference links may point in both directions, there are buttons to allow an explicit choice by the analyst to override the program.

All the cohesive annotations operate in this same general way, with minor variations on what options are available in the pop-up menu for a particular marking. For example, certain types of pronoun marking require pronoun features (such as person and number) to be indicated; in this case a button 'pronoun features' brings up a sub-menu of possible features to be marked on this instance of the pronoun.

The basic strategy of the analyst, then, is

- to work through the text from beginning to end (apart, as indicated

above, from cataphoric reference), using the mouse to select a single word or a stretch of text
- to click on one of the cohesive marking buttons to bring up a list of options
- to click on some of the option buttons and/or references in the antecedent lists
- to end with an implicit or explicit confirmation (or cancellation) of the annotation being constructed, after which it is inserted in the text window (and the antecedent list if appropriate)
- to repeat the above steps.

There are also subsidiary buttons, to provide a few other necessary requirements. A 'delete' button allows a complete cohesive annotation to be removed (and then re-inserted in the correct form in the normal way, if required). Early designs of the XANADU program called for a range of editing buttons, to correct what were expected to be common types of error, but we have never felt the need to incorporate these. A 'search' button allows simple or repeated searches on sequences of words forwards or backwards through the text. Overall control of the editing process is provided by buttons for 'quit' (that is, 'abandon the session', without retaining any of the annotations inserted this time), 'save the current state of the text file and annotations and continue the session' and 'save the current state of the text file and annotations and terminate the session'.

12.2.2 Further developments to the editing program

The previous section described the basic structure of the original design of the XANADU editor. It is also instructive, however, to illustrate the **development cycle** of interaction between a team of analysts and the software. Over the period of development of the annotation system there were a number of refinements in what is to be annotated and how it is to be labelled. The analysts' experiences of the system also led to further changes. This section describes and discusses some of the consequential design changes to XANADU and the annotation process.

a. During the original design, the expected use of the program was for the analyst to make essentially one pass through the text, marking the cohesive structures on the way through (with some local change of direction, to cope with cataphoric references and a change of mind by the analyst). For this reason the antecedents already marked were displayed in a list ordered as follows: if several items had the same index number, then the textually latest one was chosen as the 'exemplar' of that index number; and these items were then displayed in textual order, with the textually latest at the

top of the list. Thus when working at or near the end of the annotated section of the text, the analyst would usually find any required antecedent among those most recently encountered, at or near the top of the list.

This organization of the list works well in the first phase of editing. However, analysts often turned out to want to do a second pass through the text to make corrections, after thinking 'off-line' about particular annotation difficulties. Now the ordering of antecedents by their latest position within the text is often less helpful: the analyst scrolls to the appropriate part of the text window, and wants the antecedent list to show the antecedents appropriate to that section. Various schemes were proposed, but no completely satisfactory way of meeting this requirement was agreed upon. Finally a rather simple scheme was implemented: since the analyst usually knows the index number required (for example, from a hard-copy listing of the text with cohesion marks included), the program now allows the antecedent list to be displayed either in textual order, as in the original system, or in numerical order of the index numbers, which allows simple scrolling to find the required antecedent for re-editing. The analyst can change back and forth between the two orderings as required by the editing task.

b. The value has already been mentioned, for syntactic annotation (see Section 3.2), of allowing unlabelled brackets as a 'safety valve' for the analyst, so that a feature felt to be significant, but whose exact annotation is unclear, can be marked without forcing it into one of the categories provided. A similar safety valve was provided in the cohesion annotation, in the form of a comment button. The analyst could click on a point in the text and then on the comment button, to insert a free-text note (which had to be typed at the keyboard) or any one of some ten to twelve pieces of 'canned' text attached to option buttons. The analyst was encouraged to use the latter mechanism if at all possible, perhaps supplemented with a free-text message. However, the facility was flexible in allowing significant problems or departures from the annotation guidelines to be marked: these could then be searched for later, and edited into an alternative form if desired. In the most recent version of the program, the comment option buttons have been reduced to four (since another safety valve has now been provided – see (c) below: they indicate

- a possible error in the syntactic marking
- a point in the text requiring further checking
- a case where the syntactic marking of the text, though correct, precludes the required cohesive marking – this is discussed further in (d) below

- an unusual or noteworthy cohesion feature.

c. The development of the notation has now been provided for marking such marginal cohesive features as 'implied antecedents', 'inferrable *of*-complements', and 'miscellaneous cohesion' (see Sections 5.3.7–5.3.9). The guidelines for the first two of these have been drawn fairly tightly, with the result that these types of annotation have been sparingly used. However, the category of **miscellaneous cohesion** is an open-ended device, acting as an alternative 'safety valve' to the commenting facility, for cases where the analyst has no precedent for using one of the mainstream cohesion markings. It would be possible, when a suitable body of text has been annotated, to look at all 'miscellaneous' cases again, with the possibility of changing, eliminating or otherwise rationalizing them.

d. Most of the cohesion annotations are attached to a sequence of one or more orthographic units (words and possibly punctuation marks). However, the marking of **ellipsis** is special in that it is attached, not to a series of words, but to a point in the text from which it is understood that material has been ellipted. Thus, in:

John (11 was eating 11) an ice cream, and Mary <ELIP=11 a bun.

the notation indicates that the sentence is to be understood as having the antecedent *was eating* ellipted from the position marked.

There is sometimes a problem with the insertion of an ellipsis marking, since the text being marked for cohesion has already been syntactically annotated. As the syntactic annotations are not visible during the editing process, a single position in the visible text could correspond to several possible positions in the syntactic parse. Consider the sentence:

The boy (16 sat 16) in the back, and the girl <ELIP=16 in front.

Here the syntactic information has been omitted from the sentence, just as it would be in the text window. The marking for the ellipsis of *sat* lies between the words *girl* and *in*. However, the syntactic labels N] and [P also lie between *girl* and *in* (i.e. closing the noun phrase subject and opening the prepositional phrase), and the editing program cannot decide which is the correct position among such alternatives. In this kind of situation, where the program detects alternative positions for the ellipsis marking, it displays a window showing an appropriate portion of the text including the syntactic annotations, allowing the analyst to click the mouse on the appropriate position.

e. Most of the stretches of text marked with cohesion annotation are syntactic constituents, and any attempt to mark a sequence of units which violates the hierarchical structure of the parse tree will usually prove an error. Hence the program checks any sequence of units selected for annotation against the hidden syntactic marking, to ensure that the markings respect the hierarchical parse tree. When this check fails, the program displays an error message, and a window appears showing the words and the syntactic marking in the area of the attempted insertion. There is also a 'zoom' button, allowing the detailed syntactic structure of any selected sentence in the text window to be displayed for inspection by the analyst in the few cases where this is important.

There have turned out to be one or two major cases where it is essential to be able to mark a stretch of text in a way which violates the hierarchical structure of the syntactic markings. For example, an antecedent may extend over a sequence of sentences together with a partial sentence, typically in reported direct speech:

> John said, '(12 This is what I propose. We go to London tomorrow. 12)'
> He told me <REF=12 this at lunch.

This type of structure occurs so often in the texts being annotated that the program has been designed to test for it when checking the match with syntactic bracketing, and to allow it to stand.

It should be said, in conclusion, that using the XANADU editor is the key element, but not the only element, in the overall process of cohesion annotation. Other procedural phases play a notable part, including the preparatory procedures for selection and logging of texts for particular analysts, syntactic annotation, a re-editing cycle, and checks for quality control. Ideally there should also be an automatic post-processing procedure to validate and augment the human analysis.

A number of small 'filter' programs have been written to run over the annotated text to search for common problem areas. However, there are plans for a further program to be implemented to carry out such checks automatically, tidy up the annotation, and insert further markings which can be added by rule. An example of this last is the automatic marking of identical naming expressions as coreferential, thus relieving the analyst of much tedious manual annotation (although a manual check would still be necessary). Although the XANADU editor is now in a relatively stable framework and capable of efficient routine use, there are still further enhancements to be added which would speed up the whole annotation process and increase reliability.

12.3 Automatic Semantic Tagging

In the remainder of this chapter, we turn to automatic higher-level annotation, and more particularly (as an example) to semantic tagging. So far, no entirely successful semantic tagger exists, although promising developments are underway. Here we discuss the design of such a **semantic tagger**, which is probably the most attainable tool for automatic semantic annotation, just as a grammatical tagger is the most useful and manageable first-stage tool for grammatical annotation. A full-blown semantic analyser, which undertakes an integrated 'parse' of the meaning, or logical form, of any corpus sentence, seems beyond the threshold of possibility at the present time.

A semantic tagger is easy to conceive of: it is a tool which assigns the appropriate sense (or 'dictionary meaning') to each lexical unit (i.e. word or idiom) in a sentence. The task to be performed is therefore parallel to that of a grammatical tagger, although it is more abstract and more difficult. The task begins with a set of semantic categories such as the thesaurus-like set of semantic tags discussed in Section 4.6. As in the case of grammatical tagging, the task subdivides broadly into two phrases:

Phase I: Tag assignment: Attaching a set of potential semantic tags to each lexical unit

Phase II: Tag disambiguation: Selecting the contextually appropriate semantic tag from the set provided by Phase I.

For example, for the noun *spring* at least two senses (that applying to a metal coil, and that applying to a season of the year) should be attached to the word after Phase I. After Phase II, on the other hand, the program should have selected the 'season' sense, rather than the 'coil' sense, for Shakespeare's *Sweet lovers love the spring.*

However, again as with grammatical tagging, so with semantic tagging, both Phase I and Phase II are complicated by the existence of multiword units which need to be assigned a single tag. The sequence *have/get egg on one's face* can be interpreted either in terms of a sequence of semantic categories representing its literal meaning, or in terms of a single meaning representing its metaphorical, idiomatic meaning 'to appear foolish'. In semantic tagging (unlike the multiwords of syntactic tagging discussed in Section 2.2) such composite units match the layperson's notion of an 'idiom': a word sequence whose meaning cannot be predicted or derived from the meanings of its parts.

If we consider the word *spring* (and thousands of other words) it is clear that the right sense of the word can be partly determined by POS tagging,

using an automatic tagger such as CLAWS (Chapter 7). Thus, we assume there is a lexicon entry for *spring* which specifies firstly the possibility of a noun tag or a verb tag, and secondly the possibility that the noun may have the 'coil' sense or the 'season' sense:

word form	**POS tag**	**semantic tag**	
spring	noun	[season sense]	[coil sense]
spring	verb	[jump sense]	

In this sample lexicon entry, the POS tagger, by choosing the noun tag, obviously eliminates one of the senses ('to jump'). Hence the semantic tagger's task is simplified to choosing between the 'season' and the 'coil'.

Considerations like this lead to the conclusion that a POS tagger should run over the text as a preliminary to semantic tagging. In addition, POS tags can be important for the formulation of idioms: for example the idiom referred to above should be formulated as HAVE/GET *egg on* APPGE *face*, where HAVE and GET are lemmas, and APPGE is the tag representing the class of possessive determiners *his, her, my, your*, etc. In fact, there are also strong arguments that only a full syntactic annotation provides an informative enough input for semantic tagging. In view of the lack of adequate full corpus parsers at present, we may have to be content with partial parsing (for example, identifying noun phrases and their heads) as a basis for the semantic level of annotation. As an instance of the need for parsing input, we may note that many idioms cannot be properly specified without making use of categories such as noun phrase (NP): HAVE *(NP) in view* (meaning 'intend'), for instance, is an idiom which can be interrupted by a range of possible NPs, as in *has it in view*; *having several different aims in view*.

Thus, by the end of Phase I, to specify the range of sense ambiguities that a sentence may express, the following set of programs needs to have run:

- *Word tagger* assigning one grammatical tag to each word (together with one or more semantic tags)
- *Idiom finder* assigning one or more semantic tags to a whole sequence of words as a block
- *(Partial) parser* assigning constituent labels such as NP to a sequence of words, and identifying one word as the head of such constituents

We may think of an idiom finder as having the flexible search formalism of the Template Tagger (Chapter 8), with features such as the use of wild cards and regular expressions to seek out a potentially very large set of possible word sequences matching a particular formula. Adding to this the

power to recognize syntactic constituents such as noun phrases extends the power of the formalism still further.

The dividing line between Phase I and Phase II is by no means absolute: the idiom finder not only adds new meanings (as in the case *have egg on one's face*) but also subtracts existing potential meanings: for example, in the idiom *shoot the rapids, shoot* cannot have the meaning 'discharge (a firearm)'. However, now we consider more generally what procedures are necessary to eliminate ambiguities, and to arrive at the correct or most likely sense (or semantic tag) for a word.

1. *POS tag* As we have already noted, some senses can be eliminated by prior POS tagging, as in the case of *spring*.
2. *General likelihood ranking for single-word and idiom tags* One of the most powerful elements in disambiguation appears to come from general information about the likelihood of this or that sense in the language. Hence is it important to rank senses in terms of frequency, even though at present such ranking is derived from limited or unverified sources such as frequency-based dictionaries, past tagging experience and intuition. For example, *green* referring to colour is generally more frequent than *green* meaning 'inexperienced'. Some dictionaries list the first sense of *odd* as 'strange, unusual' (as in *odd behaviour*), but our experience shows that with spoken data, the 'occasional' sense (as in *odd pint*) is more common. In general, it may be advisable to determine general likelihood ranking separately for written and for spoken language.
3. *Overlapping idiom resolution* Normally, semantic idioms take priority over single word tagging, but in some cases a set of idiom templates will produce overlapping candidate taggings of the same set of words. A set of heuristics would have to be applied to enable the most likely idiom to be treated as the favourite for tag assignment. The heuristics would take account of length and span of the idioms and how much of a template is matched in each case.
4. *Domain of discourse* Knowledge of the current domain or topic of discourse could be used to alter rank ordering of semantic tags in the lexicon and idiom list for a particular domain. Consider the adjective *battered* to which three candidate tags can be assigned: 'Violence' (e.g. *battered wife*), 'Judgement of Appearance' (e.g. *battered car*), and 'Food' (e.g. *battered cod*). If the topic of conversation was known to be food, then we could automatically raise the likelihood of the 'Food' semantic tag, at the expense of the other two tags. Similarly, in a financial domain, the meanings of *bank* and *account* are likely to have the banking meaning.

5. *Text-based disambiguation* It has been claimed (by Gale *et al.* 1992), on the basis of corpus analysis, that to a very large extent a word keeps the same meaning throughout a text. (For example, if a text on one occasion uses *bank* in the sense of 'side of a river', all other occurrences of *bank* are likely to have that same sense.) If this claim is more generally substantiated, then it would be a powerful element to add to the mix of determinants of word sense.
6. *Template rules* The Template Tagger mechanism as discussed above for idiom-finding is also useful in identifying phraseological contexts in which a word is constrained to occur in a particular sense. Consider the meaning of the noun *account*: if it occurs in a sequence such as *NP's account of NP* it almost certainly means 'narrative explanation', whereas if it occurs in a financial context, in such collocations as *savings account* or *the balance of...account* it almost certainly has the meaning of a 'bank account'.
7. *Local probabilistic disambiguation* It is generally supposed that the correct semantic tag for a given word is substantially determined by the local surrounding context. To return to the example of *account*: if this noun occurs in the company of words such as *financial, bank, overdrawn, money*, there is little doubt that the financial meaning is the correct one. However, we could identify the surrounding context not only in terms of (a) the words themselves, but in terms of (b) their grammatical tags, (c) their semantic tags, or (d) some combination of (a)–(c). Hence some degree of experiment, using a training corpus and a test corpus (see Section 7.1), is necessary to determine what weight to give each of these contextual factors in selecting the correct semantic tag for a given word or word class. Other factors which need to be determined are:

   - What span of words to the left and/or to the right of the target word should be used in estimating the most likely sense? At one extreme, one might take the whole text as the 'span'; at the other extreme, our preliminary experiments indicate that the words immediately to the left or right of the target word would have a more powerful influence than other words in the immediate neighbourhood.
   - Which parts of speech should be taken into account in defining the context? Thus, it is likely that the semantic tags of 'content words' such as nouns, verbs and adjectives will be more informative in determining the semantic tags of neighbouring words than the semantic tags of 'function words' such as determiners and conjunctions. Thus, as one option, the calculation could be based on content words only.
   - What statistical technique should be used in calculating the most

likely tag? The stochastic models (especially HMM) that have been applied to POS tagging could also be applied to semantic tag disambiguation. However, given that semantic class assignment is felt to be less closely bound to word-sequence than is POS assignment, more promise appears to lie in various co-occurrence clustering techniques, such as mutual information (cf. Section 15.3). In fact, the most successful measure, from our experiments, is the Jaccuard measure.[4]

- How far it is feasible or necessary to employ syntactic information? For example, it is plausible that the heads of phrases (e.g. principal nouns, verbs, and adjectives in each sentence) would contribute more to disambiguation than dependent or modifying words.
- What weightings are to be given to alternative tags for the same word?

As in the case of POS tagging, the use of statistical methods to disambiguate senses depends on the optimization of likelihood across a whole stretch of text or (ideally) the whole text. Thus, while we have seemed to presume in our discussion above that one tag (the tag of the 'target word') is underdetermined, and other tags in the context are known, in fact the technique requires that all possible disambiguations in a stretch of text are evaluated so as to determine the most likely overall result. In a full implementation of local probabilistic disambiguation, then, it is necessary to take account of all possible semantic tags (i.e. those which have not been previously eliminated by other methods), giving them whatever preference weightings can be determined by other methods, such as general frequency ranking.

To answer all the above questions, a considerable amount of experimental research is needed. In addition, if we return to the seven methods of disambiguation listed above, there will be different views on which techniques are the most promising ones to be pursued. If we judge from the debates about probabilistic and non-probabilistic approaches to grammatical annotation (see Chapter 7), it is to be expected that (6) template rules and (7) local probabilistic disambiguation will appeal to different schools of thought. On the other hand, for semantics it appears that wider factors of context, e.g. (4) domain and (5) text, play a larger part in sense resolution, and these factors will need to be given their full weight as against local context. Different variations in the choice and combination of factors will be the basis of much further experimentation and debate in the field of semantic annotation.

Notes

1. Section 12.2 is a revised and abridged version of Garside (1993), first published in *ICAME Journal* 17, 5–27. It is republished with the permission of Stig Johansson (editor) and Knut Hofland (of the Norwegian Computing Centre for the Humanities). The author of XANADU is Roger Garside. For Section 12.2, acknowledgement is made to Steve Fligelstone and Geoffrey Leech (UCREL) and Ezra Black (ATR; formerly IBM) for their role in developing the notation; and to Jean Forrest, Elizabeth Eyes and Simon Botley for using and critiquing both the notation and the XANADU editor. We are grateful to the IBM Thomas J. Watson Research Center, Yorktown Heights, and the IBM UK Scientific Centre, Winchester, for their sponsorship of, and collaboration with, this research.
2. The AP (Associated Press) corpus consisted of American newswire stories on diverse subjects such as sport, crime and international events. Approximately 130,000 words of this corpus were anaphorically annotated to comprise the 'Anaphoric Treebank'.
3. It would be possible to eliminate the 'confirm' button, by allowing implicit confirmation by time-out or on detection of the first action for the next marking (i.e. by clicking on another stretch of text), but the analysts have not felt this to be an important improvement.
4. This measure is explained as follows. Given two events X and Y (in this case, X and Y are two semantic tags), if a is the frequency of occurrence of both X and Y in the same context, b is the frequency of occurrence of X without Y, and c is the frequency of occurrence of Y without X, then the Jaccuard measure is $a / (a + b + c)$

13

A Corpus/Annotation Toolbox

TONY McENERY and PAUL RAYSON

In the preceding Chapters 7–12, we have been concerned with **annotation software**, that is, with software intended to aid or undertake the task of corpus annotation, such as taggers or corpus editors. In this chapter we take a more inclusive view of the software needed for use with corpora, and more particularly, with annotated corpora. What tools are needed for the creation and exploitation of annotated corpora? Can we conceive of a **corpus toolbox** which would provide an overall environment for annotated corpus development and use? What in general are the functionalities and architectures of software required for this purpose? It is probably even more important here than elsewhere in this book to stress the extremely limited capability of software tools, in comparison with what we would like them to be able to do in terms of the intelligent, comprehensive modelling of language. As Church *et al.* put it (1990: 4): 'If the tools were better, computational linguists could attempt to model many more sources of constraint than they are able to deal with right now.' This complaint is not likely to disappear in the foreseeable future.

The following survey will consider corpus/annotation tools under three major categories (corpus annotation in Section 13.2, corpus editing in Section 13.3 and extraction of information in Section 13.4), and will then give attention (in Section 13.5) to three interrelated issues which cannot be overlooked in discussing corpus software: (i) single-task vs. multi-task software; (ii) modular and integrative architecture; (iii) use of general purpose text-handling tools vs. dedicated corpus-handling tools.

13.1 The Main Functions of Corpus/ Annotation Tools

It is useful to begin by laying out the major functions of corpus tools, before giving some attention to these functions separately: see Box 13.1.

Box 13.1 Functions of corpus/annotation tools

1. Corpus development (*the input of annotation information into a corpus*):
   (a) Text encoding
   (b) Annotation
   (c) Encoding of annotation
2. Corpus editing (*changing annotation information in a corpus*):
   (d) Correction (including correction of annotations)
   (e) Disambiguation of annotations
   (f) Conversion/transduction of annotations
3. Extraction of information (*the output of annotation information from a corpus, whether raw or annotated*):
   (g) Concordancing
   (h) Frequency analysis
   (i) Input to lexicons, grammars, etc.
   (j) Information retrieval
   (k) Bilingual/multilingual variants of (g)–(j)

In general, it is unrealistic to make a clean break between tools for raw corpora and for annotated corpora. In Box 13.1 (a) and (c), for instance, we have seen that encoding (mark-up) is something that applies both to the basic text and to the annotations (see Section 1.3). In this chapter, however, we focus particularly on the software developments needed for annotated corpora, and one way to express this is to say that we are especially interested in how software can be, and is being, developed to show 'basic linguistic intelligence' or 'annotation awareness'.

13.2 Tools for Annotated Corpus Development

This category of software has already been discussed in earlier chapters, and needs only cursory treatment here. We can disregard the Box 13.1 *(a) Text Encoding* software as irrelevant to our present purpose, merely noting that the growing tendency to use TEI guidelines (see Section 2.4) for the representation of the 'raw text' requires the use of software for input and validation of SGML mark-up. Similarly, we need say little further about *(b) Annotation* software, except to recall (from Chapters 7–12) the distinction between predominantly manual and predominantly automatic annotation procedures: the former represented, for example, by the tagger CLAWS (Chapter 7) and the latter by the editor Xanadu (Chapter 12). The fact that tools for manual annotation input are called 'editors' shows that the boundary between **annotated corpus development** and **annotated**

corpus editing in Box 13.1 (1) and (2) is not a watertight one. We will also note the interaction between Box 13.1 (1) and (3): automatic annotators (such as taggers and parsers) are, in effect, linguistic analysis tools, which therefore require for their operation complex **linguistic information resources** such as lexicons and grammars. These resources are themselves primary beneficiaries of the (3) **information extraction** function, and hence there may be an iterative recycling from (3) to (1): extraction of linguistic information in (3) potentially enhances the input of information in (1). It is evident, already, that the tri-functional picture offered above is somewhat oversimplistic.

The third corpus development function, **encoding of annotation**, requires the existence of tools to aid addition and validation of SGML mark-up in accordance with some standardized system such as the CDIF specification document for the BNC (see Burnage and Dunlop 1993) or the EAGLES guidelines (Ide 1996). Although one of the goals of MULTEXT[1] is to develop tools of this kind, it must be admitted that this is one area where there remains a dearth of generally available software.

Annotation encoding software cannot be discussed in isolation from the 'storage architecture' question of how to represent, in an encoded corpus, the relation between the **base text** and the annotations. It has been assumed up to now that the annotations are interspersed with the base text, as part of the same composite document. Two other arrangements are possible. One is to use the form of a **relational database**, where different fields of information represent the base text and different levels of annotation. This is particularly suitable for multilevel annotation, including, for example, POS tagging, syntactic annotation, and prosodic annotation. For precisely that purpose, it has been used by Knowles and Roach (Knowles 1995) in producing the MARSEC CD-ROM version of the Spoken English Corpus (see Section 6.1.1). No special software is needed for this application, a general-purpose off-the-shelf database system being adequate. A second alternative is to hold the base text and the annotations in separate files, with links relating each part of the one to each part of the other. This is the option favoured by Ide (1996) in the EAGLES guidelines for text representation, and also followed by Thompson and McKelvie (1996) at Edinburgh. This method allows greater freedom than interspersing text and annotations: for example, it is possible to deal with the tags for merged words such as *du* (= *de* + *le* 'of the') in French without drawing artificial boundaries within a single morpheme. Both Ide and Véronis and the Edinburgh team are working with the SGML-based TEI guidelines, and since some variant or other of SGML mark-up is increasingly being used for corpus encoding, it is likely that the need will increase for efficient handling of annotations within such an international text-encoding standard.

The Thompson and McKelvie method is to make use of what is in effect a hyperlink architecture for cross-referring between the base text and different levels of annotation. In this way, overlapping hierarchies of annotation (which can be a bugbear in SGML) can be reasonably handled. There is need for an 'SGML application development toolkit' such as LT NSL under development at Edinburgh (Thompson and McKelvie 1996), and particularly for the adaptation of such a toolkit in the direction of inputting and editing annotations. In this area, it must be admitted, suitable publicly-available tools are required with some urgency.

13.3 Tools for Annotated Corpus Editing

As already noted, **annotated corpus editing** can refer to any procedure of changing the linguistic annotations in a corpus. In Box 13.1 2(d)–(f) three reasons have been suggested as to why such annotations should need to be changed. The first is to **correct errors**, for instance errors resulting from the use of automatic annotation tools such as a probabilistic tagger. The second is to **eliminate ambiguities**, such as the ambiguities left in the text by automatic annotation tools which allow ambiguous output (e.g. ENGCG – see Section 3.3.5 – or the variant of CLAWS4 which outputs portmanteau tags – see Section 9.3). The third is to **convert** one set of annotations to another set for which there is a need: for example, it might be decided to adapt the grammatical tags of a corpus from one tagset to another which is more amenable to other users' requirements.

The term **editor** may also apply to a tool for manually *adding* annotations to a corpus, such as the Xanadu tool explained in Chapter 12. However, we will assume that this now needs no more discussion. There is a distinction, here, as with annotation input tools, between primarily automatic and primarily manual processing. The idea of a tool to edit a corpus by hand is familiar; but the idea of an 'automatic editing' tool may be less so. Chapter 8, however, described in some detail such a tool (the Template Tagger), and in Section 8.4 pointed out that it has diverse functions. As a general corpus editing tool, the overall purpose of the Template Tagger is simply to apply rules which change one set of annotations into another: such rules could either add, convert, or subtract annotations from the corpus.

Nevertheless, our main interest here is in editors in the familiar sense of tools which allow the user to change the form of a text stored in the computer. Up to a point, it is possible to rely on general-purpose text-editing software such as a screen editor (for example Emacs or even the egregious Vi). But if one is trying to correct a large annotated text or an

annotated corpus of any size, the need for a dedicated editor, which will aim to eliminate unnecessary human labour and error, soon becomes imperative. Moreover, these days much of the attention of those developing the editor will be directed to making a good graphical interface, offering the human annotator trouble-free and efficient interaction with the annotated text.

13.3.1 Manual annotation editing

In this section we discuss the facilities which would be required for a reasonable tag editor, whether the tags be of the syntactic varieties discussed in Chapters 1 and 2 or semantic as discussed in Chapter 4. Over the years we at UCREL have constructed a variety of editors which implement this list of requirements to a greater or lesser extent, perhaps the most complete being a program called Xanthippe which in one of its incarnations has been used to edit the syntactic tags of parts of the BNC and in another has been used for the syntactico-semantic tags of the ATR project (discussed in Chapter 11).

We can generally suppose that the text has been through some form of automatic assignment of tags, and there will usually be a tag indicated as the one preferred by this automatic process, together with a list for each word of the tags rejected in the context in favour of the preferred tag. It may be that the preferred tag (and the rejected tags) are fully specified, or it may be that the automatic process is capable of assigning only an incomplete tag, or a tag to only a certain level of detail – for example, the general syntactic function might be fully specified while the detailed syntactic function and the semantic function can be added only with human intervention.

The text will typically include mark-up to indicate at least the main subdivisions of the text. In UCREL this would always be in some form of SGML. In cases where the text makes use of some other form of mark-up, we have generally found it most useful to convert it to SGML making use of some combination of general-purpose SGML parsers (such as the public-domain SGMLS) and special-purpose one-off programs built out of a standard SGML filter template.

Since a wide variety of tagsets may be in use, a tag editor needs to be written as far as possible in a tagset-independent way. It is generally possible to have a tag editor read in the tagset to be used, but we have sometimes found it necessary, because of idiosyncrasies in the tagset, to have a small amount of special-purpose code to deal with them – an example is given below.

The user interface, of course, needs careful consideration. We at UCREL

have tended to work with a small team of highly-trained corpus analysts who prefer an interface which minimizes the number of keystrokes and screen redrawing for the more common functions, even if these lead to different procedures for what are conceptually similar tasks. This orientation is also apparent in the discussion of the Xanadu editor in Chapter 12. If we had employed a larger number of less highly-trained analysts in a less intensive way, it is possible that the user interface design criteria would have been rather different, with more commonality of procedure and more prompting from the editor.

We have found the most useful screen format to be a series of parallel vertical columns containing (for a stretch of text) the words in one column, the preferred tags in another column (or perhaps sub-divided into a column containing that part of the tags fully specified by the automatic tagging system, and a second column containing that part only partly specified by the automatic tagging system), and further columns containing the rejected tags (or perhaps only the fully specified parts of these, to save screen space). There will often be further information associated with the words; for example, a reference code for the word; information about the automatic tagging process, including an indication of where the analyst's attention should be drawn to places where the process is likely to be at fault; information showing how multiword units (see Section 2.2) are linked together; and so on. It perhaps requires some comment that we generally display the words of the text down the screen, where one might expect that a horizontal display would best mimic the normal process of reading. But this is perhaps an area where the design of our user interface is influenced by the background of our analysts, since this screen format in fact mimics the hardcopy listings that most of them are familiar with from earlier projects.

A problem with this representation is that we can display only a small number of words of text at a time (perhaps twenty or twenty-five, given the size of display screen and the choice of a font size large enough to avoid eye strain). We therefore usually display a subsidiary window onto a larger stretch of text, including the words displayed in the main tagging window, but displaying only the words and none of the associated material. As the analyst scrolls or otherwise moves through the text in the main window, this subsidiary window moves through the text in a linked way.

Typical user functions for a tag editor would include the following:

(a) The promotion of one of the rejected tags so that it becomes the preferred tag, by far the most common type of tag correction for a reasonably competent automatic tagging process. For situations where

the correct tag is not among the rejected tags a panel of tags can be displayed for user selection of the appropriate one.

There can be a problem here of the size of the tagset – the BNC C7 tagset (some 146 tags) is somewhere near the upper limit while retaining an adequate font size. We would have had a problem adapting the Xanthippe editor to a tagset of the size discussed in Chapter 10 (475 tags) – a possibility would have been to divide the tagset into, say, five panels of roughly associated tags, and attempt to predict the panel most likely to contain the desired tag from the automatically preferred tag, allowing the user to select any of the other panels with a single mouse click. In tagsets we have had to deal with we have been able to divide the tagset into a primary portion (containing no more than about 150 tags) and a set of possible continuation sequences. These continuation sequences have been dividable into a small number of panels, such that the Xanthippe editor can predict the panel of continuation sequences once the primary part of the tag has been confirmed (see Section 11.4.2). This is an area where we have been forced into providing some hard-wired code to deal with particular tagset idiosyncrasies.

It would be possible to add a 'help' facility to display on-screen information about a particular tag at the analyst's request (for example, tag descriptions or guidelines for its use), but we have not felt the need to incorporate this into the Xanthippe editor. What we have generally done instead is to provide on-screen access to all information sources relevant to the tagging process, including all recently tagged texts and consolidated listings by word and by tag, through a separate program running in a separate window.

(b) Correction of the original words of the text. In some cases there is a requirement for the insertion into the text of a note specifying the original word, the corrected word, and an optional comment by the analyst. More generally there may be a need for the analyst to be able to insert some form of comment about a local aspect of the tag correction process; this is related to the idea of a 'safety valve', mentioned in Chapter 5. The editor generally inserts this comment into the text surrounded by a suitable SGML mark-up sequence, and indicating the identity of the analyst making the comment.

(c) The insertion or deletion of markings for a multiword unit. Deletion can be done with a single mouse click on any part of the marking. For insertion we have the analyst click on the first word of the sequence, and then select a length from a pop-up menu. Since we have had to deal here with the marking only of contiguous multiword units, this is sufficient. In inserting or deleting a multiword unit, and in the similar

functions of splitting a word into two or combining two words into one (perhaps because of typographic errors in the original) Xanthippe makes some rather minimal attempt to guess a suitable tag or tags for the result.

(d) If Xanthippe is being used for correcting syntactic tags assigned by an automatic process, it is likely that occasionally an automatically assigned sentence break will need to be suppressed or a new one inserted. So user functions need to be provided for this. Since this type of correction is often in the vicinity of a putative punctuation mark, it is often possible for an editing program to make some sort of informed decision as to what to do with this punctuation.

It is useful for the analyst to be able to search the text, from the present text position or from the beginning of the file, either once-off or repeatedly, for words, parts of words, word sequences, word reference codes, or even tags or tag sequences, whether fully specified or not, and given that it is not clear in this latter case whether searching *all* tags or only the preferred tags is likely to be the most useful option.

A very useful extension to this is **global editing**. If the analyst detects a persistent pattern of error in a file of text it is useful for them to be able to specify a pattern of words or tags to search for and a preferred tag or tag sequence to be applied throughout the file, with or without user confirmation of each matching instance. Xanthippe implements a fairly restricted form of pattern matching for global editing; a fully developed version of this would of course be equivalent to an interactive form of the Template Tagger discussed in Chapter 8.

If a large amount of text is being corrected from a single domain it is often possible, after a certain amount of experience of the domain, to generate lists of word patterns (particularly naming expressions of various kinds) and their associated tag sequences. We have provided our analysts with a separate tool to do this before starting the tag editing task proper with Xanthippe, using a list of phrases to be corrected which can be built up by the analysts themselves.

Finally we can log the process of manual annotation. We can simply write to a log file a list of all the tag corrections or other revisions made by the analyst, together with suitable global information as to the name of the analyst, the file being processed and the date. This is useful for extracting patterns of persistent error by the preceding automatic tagging process.

13.3.2 Automatic annotation editing

We have already made reference to the use of the Template Tagger as an

automatic annotation editor, and here this function may be briefly illustrated, using a grammatically tagged corpus as an example. It has not yet been sufficiently realised that a tagset needs sometimes to be adapted. It may be that a tagset devised (partly) for ease of automatic tagging will later prove ill-adapted to a user's needs. For example, many tagsets, including C5 and C7, do not represent auxiliary verbs as a separate category in English: something that many users may find desirable. For this purpose it is necessary to enrich the tagset by introducing new tags (for example, instead of VBZ for *is*, it will be necessary to devise two tags, say VBAZ and VBZ, for *is* as an auxiliary and *is* as a main verb respectively). To do this and other similar changes, a set of Template Tagger rules are needed enacting such changes as

> 'If a tag VB* is followed by a tag V*, with or without the intervention of other tags XX or AV0 (i.e. *not* or adverb), then change VB* into VAB*' (see Section 8.2.1).

In fact, a comparatively small number of such rules will make the necessary change, with few exceptions. This is a relatively straightforward global edit, whereas to make other changes – say, adapting the tagging of *-ing* words in order to conform to one set of guidelines rather than another – is likely to be more complex. There are other reasons – apart from the needs of a specific user – why it might be desirable to adapt annotation systems. One is to convert annotations into a form which is conformant with a externally devised standard, such as the EAGLES standard for morphosyntactic or syntactic annotation (see Chapter 16). One of the plans of UCREL at the moment is to adapt the BNC grammatical tagging to the EAGLES guidelines, thus making the tagset more directly comparable with those to be used for other European languages.

13.4 Extraction of Information from Annotated Corpora

We turn now to **search and retrieval** software for extracting information from annotated corpora. This kind of software is more familiar to the general corpus user than any other: anyone who wishes to make use of a corpus is inevitably going to look for means to extract linguistic information from it. The 'naive user' is likely first to encounter a corpus through a **concordancing** facility, that is, a program for listing (a subset of) the instances of a given linguistic phenomenon (typically a word) in the corpus, together with their immediate preceding and following context.

Associated with the concordancer will often be other facilities, providing frequency lists of word types, listing collocations based on mutual information or other measures, and furnishing information about subdivisions of the corpus, together with the incidence of linguistic phenomena in these. Many packages of this kind are available, some more advanced than others, and each tending to have its own special features. For a discussion of some of these packages, see Hofland (1991) or Kirk (1994), and for more general surveys, see Lancashire (1991) and Hughes and Lee (1994).

The functionality and usability of search and retrieval packages have been enhanced over recent years to the extent where a number of different functionalities can now be looked for:

- A Windows interface, allowing mouse-driven queries
- Pre-indexing of corpora, enabling fast on-line searches of very large corpora (up to 100 million words or more)
- Location referencing of retrieval instances, in terms of their text type, bibliographical details of written texts, or information about speakers in spoken discourse, etc.
- Extraction of concordances, frequency lists, etc. from user-defined subcorpora
- Extraction of collocations and other types of word-combination
- A flexible query syntax, which enables the user to specify the class of strings to be searched for in terms of Boolean operators, wild cards, variable intermediate gaps, etc.
- Adaptability to the recognition of different encoding formats: e.g. SGML and non-SGML based.

Choosing one package over another involves decisions about machine type, as not many packages are supported across the main platforms (UNIX, PC DOS, PC Windows, Apple Macintosh). One way to solve this in the future will be access via the World Wide Web (possibly using the Java language) which is client-machine independent. Pilot concordance services using the multilingual CRATER corpus have been provided at Lancaster University and at Birmingham (see Appendix I for addresses).

However, our main interest in search and retrieval packages in this chapter must focus on their treatment of annotations. Using a package with **annotation awareness**, it will be possible to search on annotations up to a point, but the output (in the form, say, of a concordance) is likely to be littered with annotation, with the result that no normal human user would find it easy to interpret. Hence, one useful facility is for the software to recognize annotations, and optionally to mask them from the screen interface. An example of an annotation-aware corpus exploitation tool is SARA (decribed below), and similar facilities are provided by XKwic

(Christ 1994) and ICECUP (Quinn 1993). WordSmith (Scott 1996) also offers a tag-aware facility.

13.4.1 SARA as a corpus exploitation tool

In this section we briefly discuss the SARA program as an example of a tool for the exploitation of a corpus. SARA (SGML-Aware Retrieval Application) is a program, developed by Tony Dodd as part of the BNC project, to allow a person to extract relevant information from the British National Corpus. More details about SARA, and an extended tutorial for the software, are given in Aston and Burnard (1995).

A number of general points need to be made about SARA:

(a) the size of the British National Corpus (one hundred million words, together with part-of-speech tags and associated information) places a certain number of constraints on how it can be used. First, an index has to be created from all the words of the corpus, and this is used to speed up the search for particular words or phrases. But a consequence is that certain types of query, which match the way the index is organized, are fast in operation, while other types of query, which do not match the way the index is organized and therefore have to be done largely on a word-by-word basis, are much slower. A second consequence is that, in order to allow use of the system from a small computer, it is possible to run the SARA software over a network. Thus, the corpus itself, the index, and a search engine would run on a large machine, such as the one at OUCS (Oxford University Computing Services), while the user interacts with a small local machine linked to the larger machine. A query will be built up on the local machine, the query submitted to the search engine for matching against the BNC, and the results will be sent back to the user's machine, for inspection or further processing.

(b) as has been mentioned in earlier chapters, the structure of the BNC is marked up using the SGML language. The SARA software has been designed to be aware of this mark-up, and provides for the user to specify their query in terms of the SGML structural elements specified in the BNC. Thus a query can be limited to particular elements, or a user can browse the BNC making use of the SGML mark-up as a set of signposts.

(c) unlike the Xanthippe program discussed earlier in this chapter, which was designed to be used continuously by a highly-trained analyst, the SARA software is designed for the person who makes use of it only intermittently, and who therefore will require guidance in the appropriate series of actions to carry out a particular task. SARA

has been designed to run under the Windows operating system, making use of its standard set of window layouts and menu formats, and provides extensive on-line help facilities for the user.

The basic mode of using SARA is

- to formulate a query
- to search the BNC using this query, and
- to inspect the resulting list of text samples from the BNC which match the query. It may be that this list of samples is the answer required, in which case the task has been finished with this interaction with SARA. More likely, the user will wish to adjust the query to add more samples or to eliminate some samples if too many things matched the query.

The simplest SARA search query would be to search for a single word, for example the word *umbrella*. SARA responds with an indication of how many times this word appears in the corpus, and how the occurrences are spread throughout the corpus. No definite conclusions can be drawn about how a word is used, if it mainly occurs in a small number of the text samples of a corpus. A limit can be specified on how many samples are to be displayed, and SARA can display all the samples met until the limit is reached, or one sample per text, or a random selection. For each sample SARA displays the matched word together with the contexts (a certain number of words on either side of the matched word, or the sentence or other structural unit containing the matched word). As indicated earlier, each word of the BNC has been marked with its part-of-speech, and there is a display option to colour the displayed words with reference to their part-of-speech.

An initial formulation of a query may result in too many matches being recorded. There are various ways of dealing with this. The user can return to the original query and modify it, perhaps by specifying further constraints which must be satisfied by the query. Clearly a well formulated query to solve some particular problem is a useful object in its own right, and SARA provides facilities for naming and storing queries, so that they can be recovered and reused, perhaps with further editing. Alternatively the use can 'thin' the results of a query in several ways:

- the user can simply go through the results, marking each result in turn as to whether it is an appropriate sample, and then all the inappropriate samples can be removed with a single key depression. Alternatively, the user can mark all the samples which they wish to reject, and then eliminate them.
- common words can often appear as several different parts-of-speech, so another way of eliminating inappropriate samples is for the user to

specify which parts-of-speech they are interested in. Suppose for example that all occurrences of the word token *watches* had been extracted. SARA could be requested to list all parts-of-speech occurring assigned to this word in the BNC (for example plural common noun and third person singular form of verb). The user could then instruct SARA to display only those samples where *watches* has the preferred part-of-speech.

After a set of results has been generated by SARA, a 'bookmark' can be inserted marking this set of results, so that the user can return to it another time, without having to regenerate it. SARA can sort the results (for example on the words and then on their left context, if we were particularly interested in the types of context that precede a particular word). SARA can print a copy of a set of results, for off-line inspection, or the results can be written to a file, so that they can be read into a word-processor or can be processed further by some other piece of software.

So far we have assumed that a query is basically a single word. But it can be much more complex than this:

- Instead of searching for a word, the user can search for a word pattern using a regular expression. This would allow searching for words with either a British or American spelling (for example *hono(u)r*), or for words commencing with a particular sequence of characters (such as morphological variants of a single stem). In principle it is possible to search for all words with a particular word-ending, such as words ending with the sequence *ing*, but the BNC indexing system makes this type of search inconvenient, particularly with such a common ending as this.
- the user can search for a sequence of specified words, where we allow alternative words at certain positions in the pattern, and some positions may be completely unspecified. Thus SARA could search for the pattern consisting of the words *input* and *output* (in that order) with any single word in between them.
- the user can specify what the SGML context should be for a pattern. Thus, if we were particularly interested in the structure of newspaper headline language, we could search for text patterns in an SGML 'head' element.

Complex queries can be built up in two ways. There is a query builder which uses an intuitive graphical interface to allow the user to specify a set of constraints and how they are to be linked together, by conjunction (both constraints must be satisfied) or disjunction (one or other of the constraints must be satisfied). Alternatively there is a special language (CQL,

for Corpus Query Language) into which all SARA queries are in fact translated, and the user can write complex queries directly in this language.

So far we have discussed only one mode of using SARA, namely 'Query' mode, in which the user formulates or modifies a query, and SARA returns a set of resultant pieces of text which match that query. There is another way of using SARA, namely 'Browse' mode, in which the user can utilize the SGML elements into which the BNC is subdivided to scan through a section of text. In this mode, the user begins with a list of the top-most elements into which the BNC is divided (actually the texts making up the corpus, each a 'bncDoc' element). If the user selects one of these and requests it to be expanded, SARA displays the next level of SGML elements, usually 'header' and 'body'. If the user selects 'body' and has this expanded, SARA shows the next level of SGML element, perhaps chapters or paragraphs. In this way the user can work down to the sentences and then to the words of a text of interest.

It should be mentioned, too, that the power of constructing complex queries is available in the Template Tagger (Chapter 8), which can be adapted to use as a retrieval tool. Further advances in search and retrieval packages will no doubt take into account more abstract levels of annotation, such as parse trees. The Nijmegen software Linguistic DataBase (LDB) is specifically designed to handle syntactic annotation, including extraction of trees which conform to particular retrieval requirements (van Halteren and van den Heuvel 1990). It can be predicted that such tools will become more widespread as more corpus annotations at various levels become available.

13.5 Tools, Tasks and Architectures

In Sections 13.2–13.4 we have focused on the different functions which corpus tools fulfil. It is now time to adopt a formal rather than functional perspective: what of software architecture?

In the early days of corpus software development, the typical case was a program designed and written 'in house' at the users' institution, intended to perform a single task. Naturally enough, some of this software became widely used and distributed and provided a model for further software developments. A 'corpus workbench' consisting of a group of programs was the next development. As a suite of programs having interrelated functions, the CLAN software was written (by Leonid Spektor of Carnegie Mellon University) originally for use with the CHILDES database (MacWhinney and Snow 1990, MacWhinney 1991). A more advanced cluster of the same general kind is the Lexa software suite developed by

Hickey (1993a, 1993b), which includes corpus pre-processing, annotation, and text retrieval. These 'toolkits' take quite a significant step from uni-functional to multi-functional software development, the latter also illustrated by Brodda's (1991) PC Beta software.

After the move from **single-task** to **multi-task** software development, the next logical step is to aim for modular **integrative architecture**. The development of tools to build and exploit corpora which may run to hundreds of millions of words is an expensive task in terms of time and money. It is hardly surprising, therefore, that concepts such as **re-usability** have been adapted to the field of corpus-based language engineering from the field of software engineering. A useful metaphor here is 'software Lego'. Programming practices should allow small programs to be slotted together to form larger and altogether more useful programs according to need. Developing software for new functions then need not require going back to the drawing board: a couple of pieces of 'Lego' to fit to the existing architecture may be all that is required. Two initiatives which have this modular type of design are (a) the MULTEXT project which has been undergoing development in Europe as part of the European Union's language engineering programme,[2] and (b) the GATE architecture (Cunningham *et al.* 1996) developed at Sheffield in the UK. In the MULTEXT work, as in related work at Edinburgh (Thompson and McKelvie 1996), the unifying principle is that a text stream in a standard (SGML-based) format should be pipelined between any one module and another without hindrance.

To conclude this survey on a less innovative note, it may be pointed out that considerable advantage can be gained by using UNIX facilities or off-the-shelf software such as commercial database packages. We mentioned one instance of the latter in Section 13.2, in discussing corpus storage: the database of the Spoken English Corpus. The database architecture has the advantage of using the fast indexing and data management functions already available in a commercial database package. Not all the software in the corpus toolbox has to be developed for, and dedicated to, corpus-based research.

Notes

1. See the MULTEXT/EAGLES documents (Véronis 1996; Véronis and Ide 1996).
2. See Appendix I for MULTEXT's www address.

14

A Corpus-Based Grammar Tutor

TONY McENERY, JOHN PAUL BAKER
and JOHN HUTCHINSON

14.1 Introduction

One of the fields of application in which corpora are beginning to be exploited is an educational one: that of language learning (see Fligelstone 1993, Wichmann *et al.* 1997). In this chapter we examine the potential of corpus annotation in this area, by focusing on one particular piece of software: the Cytor grammar tutoring package. It is found that teaching basic grammatical word classes by an automated method is more successful than teaching by human tutors. Moreover, compared with other computer grammar tutors, a corpus-based program has the advantage that textual material for the analysis can be varied at will.

At Lancaster, we have been as aware as any British university of the changing nature of the ambient level of grammatical knowledge amongst our students. As the teaching of grammar left the school curriculum, so the basic knowledge of grammar brought by a student to university declined. This decline is not necessarily to be regretted in itself. But for those students who clearly would benefit from a knowledge of grammar, e.g. students of linguistics, or those students for whom a knowledge of grammar may improve their ability to learn another language, this lack of knowledge is undoubtedly a handicap. Unfortunately, coupled with this unawareness is often a profound dislike of grammar – in the UK, grammar remains distressingly unpopular with students of linguistics and language, as noted by Steel and Alderson (1994).

A real problem faces the teacher here: how does one proceed to teach grammar, as one must to those students requiring such knowledge, if the students are resistant to being taught? It was to solve just this problem that work began in 1992 to develop a corpus-based, self-access grammar program. The work was initially undertaken on a shoestring by Tony McEnery and colleagues (reported in McEnery and Wilson 1994), but was later greatly aided by the award of two Innovation in Higher Education

(IHE) awards by Lancaster University. With the help of the IHE initiative, a pilot system has been developed and evaluated. The aim of this chapter is to show how, moving from this initial problem, we have worked towards a solution, guided by the existence of annotated corpora. The chapter begins with a description of the program, and moves on to a discussion of experimental evidence supporting the effectiveness of this approach. The chapter concludes by examining some insights the program provides into the difficulties experienced by students (at least, British students) in learning grammar.

14.2 The Program

The Cytor program has been developed as a PCDOS based program, designed to run on even the humblest of platforms. The mechanics of the program are relatively simple – the program reads a morphosyntactically annotated and treebanked corpus, hides the annotation, and asks the user to annotate the bare text. The user then proceeds to annotate the text, and the machine judges the accuracy of the annotations introduced by using the hidden annotations as a 'crib-sheet'. Note that the existence of manually corrected annotated corpora is essential to this program. Without them this approach to computer-aided instruction would not be possible. If we were relying on purely automatic part-of-speech and constituent structure assignment, the decline in accuracy we could expect would render this whole process pedagogically unsound.[1]

Along the way the user has access to a variety of help systems:

- The user can check the system of mnemonics that the program uses – e.g. they may enquire as to what NN1 represents.
- The user can also access a part-of-speech annotated lexicon, to check on both what parts of speech a word may have, and also how frequently the word occurs with each individual part of speech.
- The user can access an on-line concordance program to look through a reference corpus.
- The user can access an on-line phrase structure rule browser, which accesses a reference treebank to give the user an idea of typical grammatical constructions.

Users have unconstrained access to all of these help systems during the process of annotation, and, in performing the annotation, are encouraged to self-correct. The program does not give prompts or clues, other than by providing yes/no answers to proposed annotations. For any given annotation a user is allowed four chances to get it right.[2] Where the user fails to

get it right within four guesses, the program gives the right answer, and provides a brief description of the part of speech or phrase-structure bracketing that the user mis-identified.

For the teacher, the program does two distinct things. Firstly, it keeps a running log of what the user is doing – allowing a teacher to monitor how often help systems were accessed by an individual student, for example. The program also records errors made by the student, allowing the teacher to build up a view of what parts of speech tend to be difficult and/or confusing for this or that student.

Having presented this brief overview of the operation of the program, it seems reasonable now to see how the program performs in the classroom.

14.3 Experiments

The program was tested at Lancaster University and Queen's University Belfast in the period 1994–96. To give the first Lancaster experiment as an example (for a full report see McEnery *et al.* 1995), seventeen students were recruited from the English Language Part I degree scheme at the University. This was an attractive population to experiment upon, as educational attainment was removed from the set of variables. All of the students had recently achieved at least grade B at English Language 'A' level, giving a group that was well-balanced both in terms of academic ability and exposure to linguistic concepts/terminology. The students received a small fee for participating in the experiment.

At the beginning of the experiment, all seventeen students were given a one-hour lecture, going over basic part-of-speech distinctions such as noun, verb, adverb and adjective. After this the students were split into two groups balanced for gender and computer literacy/illiteracy. One group worked with the tutor program, and the other attended a weekly workshop on part-of-speech analysis.

The weekly sessions for both groups spanned five weeks and lasted for two hours.[3] The group using Cytor worked in a computer laboratory in the Department of Linguistics, and were the only students allowed access to the lab during the workshop sessions. The students had a machine each, but were encouraged to discuss the experiment with one another. A tutor was on hand to answer any questions that the students had. The human-led workshops, on the other hand, followed a workshop format, conducted by tutors experienced at teaching students at this level. During the second hour the students were allowed to work in groups, with reference grammars to hand and input from the human tutor whenever they required it.

The texts that the students worked on were taken from the Canadian Hansard corpus. This corpus was chosen for the main study as it was in plentiful supply and already accurately annotated.

The students were asked to introduce parts of speech into the texts,[4] and their responses were assessed against the answers encoded in the source corpora by grammarians at Lancaster (for a description of this latter operation see Leech and Garside 1991).

Both groups used a simplified tag set for the first two sessions (see Table 14.2, page 215). Following that, the students used the full CLAWS2[5] tag set for the final three weekly sessions of the experiment. To support them in moving to the full tagset they were given the list of tags and definitions at the end of the second session, and readings were suggested from *English Grammar for Today* (Leech *et al.* 1982) and *University Grammar of English* (Quirk and Greenbaum 1973).

At the end of the first two sessions, the students were given ten sentences of the Canadian Hansard corpus to annotate with part-of-speech information, to be completed at home before the next session (referred to hereafter as the Home Test). The following week the students started to use the full CLAWS2 tag set, and continued to use it for three weeks. After the final session, all students attended a one hour test, in which they were asked to do a part-of-speech analysis, using the CLAWS2 tagset, on paper (referred to hereafter as the Class Test). It was emphasized that they should work at a pace at which they felt comfortable: this was not a grammatical annotation race.

14.4 Results

It was hypothesized before the experiment had begun that the human-taught group would tag more words correctly than the computer-aided group, and while all subjects would become more adept at tagging over time, the human taught group would always remain slightly ahead. Figure 14.1 summarizes the mean percentage of correctly tagged words across time for both groups. In Figure 14.1, the mean accuracy attained by each group is plotted against the learning sessions T1–T7. In the first two weeks our hypothesis was confirmed: those in the human-taught group performed slightly better on average, though not significantly. However, by T3 (the Home Test) the computer-aided group had overtaken the human-taught group in terms of accuracy, and, although both groups did poorly at T4 with the introduction of the more complex tag set, the computer-aided group continued to improve while the human-taught group's performance actually declined in accuracy from T5 onwards.

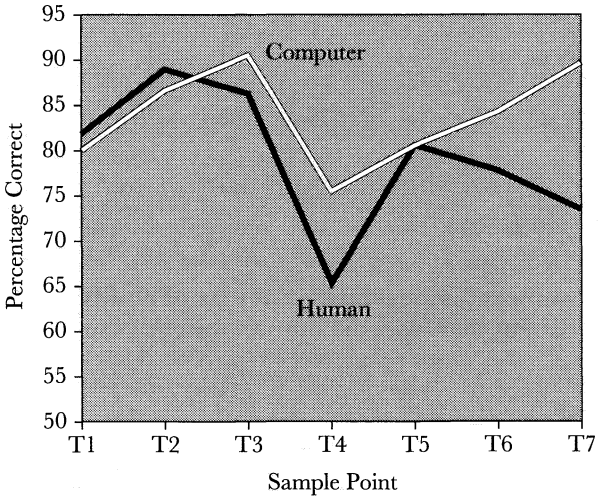


Figure 14.1 Mean percentage of correctly tagged words over time for the computer-aided and human-taught groups

Table 14.1 (overleaf) shows group performance means over time. The top number in each cell is the mean per cent correct for that week (T*X*). The bottom number represents the mean number of words attempted (whether correct of not) for that week. Note that because of the nature of the Home Test (10 sentences to be completed in their own time) the number of words attempted is identical for both groups.

At the end of the test that marked the close of the experiment, the subjects were asked to answer four questions about the study. The first three had the format of 'How difficult/interesting/useful did you find the task?' and were answered on a 5-point Likert Scale, ranging from, e.g., *Very difficult* to *Very easy*. The final question was 'Would you be prepared to take part in a similar experiment but this time for no money?' Aggregate scores were found for each subject, by the following method:

Difficulty Score (1–5) + Interest Score (1–5) + Usefulness Score (1–5) + Repetition Score (Yes scored 1, No scored 0)

A Mann Whitney test (using ordinal scales of measurement for two unmatched samples) was carried out on the questionnaire data ($U = 58.5$, $p < .05$ (one-tailed)) suggesting that overall, **the computer-aided group rated their task more positively than the human-taught group.**

One final point is worthy of note. The standard deviation from the mean for the human taught group was 7.447, while this figure was 2.306 for the computer taught group. This shows that **the computer-aided**

Table 14.1 Group means: quality and quantity of performance

Group	No.	T1	T2	T3[a]	T4	T5	T6	T7[b]
Human	9	81.76	89.01	86.27	65.20	80.82	77.80	73.64
taught		393.1	405.7	276.0	107.4	102.4	256.8	349.3
Machine	8	80.17	86.89	90.30	75.73	80.82	84.49	89.34
taught		894.0	119.0	276.0	349.9	690.0	826.3	448.6

[a]Home Test [b]Class Test

group was more homogenous in terms of ability at the end of the experiment than was the human-taught group.

14.5 Discussion

The first thing that must be stated is that the results presented here have been replicated with a further cohort of students at Lancaster, and a cohort of students in the English department at Queen's University Belfast. The 'superhuman' efficacy of the computer-aided grammar instruction seems to be undeniable. When we consider the impact of the results outlined, their significance becomes intriguing.

The most notable finding that the results immediately presented was that our initial hypothesis was incorrect. As noted, the computer-taught group eventually clearly out-performed the human-taught group. So Cytor, in this experiment, produced a set of students who could carry out the required grammatical annotation to a higher degree of accuracy than those students taught by expert human grammarians.

A second point can be made about the nature of that accuracy. The students in the computer group were all of a similar level of ability at the end of the experiment. The students in the human-taught group varied greatly in their individual ability. Not only did Cytor produce students which, as a group, were accurate annotators: as individuals also all of these students benefited roughly equally from the interaction. The same cannot be said of the human-taught group.

Studies which claim that new approaches are 'more interesting' or 'more exciting' are, and should be, treated with scepticism. The number of factors which can contribute to one set of students, in one place, at one time, with one tutor deciding that some system or course is better than another are so many that it hardly seems worth commenting on the attitude questionnaire at all. Nevertheless, here the students were evaluating a system which seemed to force them to indulge in a great deal of what students have traditionally found distasteful: grammatical analysis. The fact that the human-taught class were so negative in their evaluation came

as no surprise. But the fact that students who had worked in a solitary fashion on an unpopular subject for a great deal of time were so positive does seem remarkable. Hence, while not suggesting that a panacea for the public re lations problems of grammar as a subject has been discovered, this result at least seems to imply that a self-access, non-judgmental, instant-feedback approach to the teaching of grammar may serve the student well. Only time and further detailed experiments will determine how far the computer itself is an integral or an incidental part of this process.

So to summarize, the Cytor system seems to have produced students who were more accurate grammatical annotators as a group, more homogenous in terms of their learning experience, and more satisfied with the learning process than those students taught as a group by human grammarians. The ability of the program to act as a reliable instructor, because of the manually-annotated corpus it uses, must be crucial to these results.

14.6 What Do We Learn About the Process of Learning?

We have seen that a corpus-based approach to computer-aided grammar teaching is attractive: it is worth considering now what is revealed about the nature of learning grammatical distinctions by the log of errors that

Table 14.2 The simplified part-of-speech tagset

Tag	Description
$	possessive (*'s*)
A	article (*the, a, an* etc)
B	before conjunction (e.g. *in order*) or infinitive marker (e.g. *so as*)
C	conjunction (*and, or, but, if, because*)
D	determiner (*any, some, this, that, which, whose* etc)
E	existential *there*
I	preposition (*of, on, under* etc)
J	adjective (*fat, young, old* etc)
L	leading coordinator (*both* in *both..and, either* in *either..or* etc)
M	number or fraction (*two, three, 10's, 40–50, tens, fifth, quarter* etc)
N	noun (*dog, cat, books, cats, north* etc)
P	pronoun (*he, him, her, she, it, his, everyone, none* etc)
R	adverb (*else, namely, very, how, more, alongside, where, longer* etc)
T	infinitive marker (*to*)
U	interjection (*oh, yes, um* etc)
V	verb (*is, was, had, will, throw* etc)
X	negator (e.g. *not*)
Z	letter or letters of the alphabet (*a, b, c* or *a*s, *b*s, *c*s etc)

the program yields for each user. We will focus initially on the simplified part-of-speech tagset here, as its size makes the presentation of results more manageable. This tagset, based on the first characters of an earlier version of the C7 tagset (see Appendix III), is summarized in Table 14.2. Table 14.3 illustrates which parts of speech were mistagged as other parts of speech in what volume week by week. (We have retained only significant categories of error in this table. Note that for purposes of this comparison, all of the tagging decisions of weeks three, four and five have been mapped on to the simplified part-of-speech tagset. This causes within-category errors mainly from week three onwards, e.g. proper noun misclassified as common noun. The table represents the errors as average numbers of errors per thousand tagging decisions.)

In order to interpret these errors, we performed a one-tailed Pearson correlation test. The questions we wanted to ask are as follows:

1. Do the students get better at making distinctions across the board as the weeks go by?

Table 14.3 Frequent mistaggings recorded by the automated grammar tutor

X mistagged as Y	Week 1	Week 2	Week 3	Week 4	Week 5
A–A	–	–	4	1	2
A–P	12	4	2	1	1
C–C	–	–	1	2	3
C–I	6	5	4	5	3
C–R	3	1	0	3	2
D–J	2	4	4	2	1
D–R	3	3	0	1	1
I–C	6	2	1	4	1
I–I	–	–	12	6	5
I–R	3	3	1	2	3
I–T	2	1	1	2	2
J–J	–	–	3	6	5
J–N	9	9	8	9	6
J–R	2	2	0	4	1
J–V	2	4	1	2	2
N–J	4	3	7	3	4
N–N	–	–	65	26	30
N–R	0	0	1	1	1
N–V	3	5	2	3	3
T–I	2	2	0	0	1
V–J	3	1	0	0	1
V–N	2	1	0	2	1
V–V	–	–	30	20	16

2. If not:
   (a) what are the distinctions that improve?
   (b) what are the distinctions that fail to improve?
   (c) are there any distinctions which students actually get worse at making?

The results of the test are shown in Tables 14.4 and 14.5. Table 14.4 lists the within-category errors, while Table 14.5 lists the across-category errors. In interpreting the results, the closer the correlation to 1 or −1, the better the correlation. The closer to 0, the weaker the correlation. A negative correlation shows that error rate is decreasing over time. The one-tailed Pearson test, listed as the p value in the tables, is used to test where items are significant at the 10 per cent level – those that are have p values of .10 or less. Values at .10 and below have been starred.

Table 14.4 Within-category errors

Pair	Correlation	p value
A–A	−.982	.061*
C–C	1	.000*
I–I	−.924	.125
J–J	.654	.273
N–N	−.815	.196
V–V	−.977	.077*

Table 14.5 Between-category errors

Pair	Correlation	p value	Pair	Corelation	p value
A–P	−.85	.033*	J–R	.00	.50
C–D	−.73	.076*	J–V	−.29	.32
C–I	−.83	.040*	N–J	.00	.50
C–R	.50	.50	N–R	.87	.03*
D–J	−.47	.21	N–V	−.29	.32
I–C	−.58	.15	T–I	−.61	.14
I–R	−.18	.39	V–J	−.65	.12
I–T	.29	.32	V–N	−.19	.38
J–N	−.73	.08*			

14.6.1 Within-category errors

If we consider our first question (Do the students get better at making distinctions across the board as the weeks go by?) the answer is that on the

whole they do, but with one notable exception. There are three categories in which the p value indicates no significant changes of error rate, but the other three categories of error do show a significant change of error. The negative correlation for A–A and I–I show that these distinctions improve over time. The positive correlation (of 1) for C–C, on the other hand, shows the reverse tendency.

The distinctions that decidedly improve are the A–A and V–V distinctions. The I–I, J–J and N–N distinctions fail to show a **significant** improvement. Of these the I–I and N–N distinctions show a trend towards improvement, while the J–J distinction actually shows a trend towards deterioration. These last three observation are, however, somewhat weak as the correlations observed are not significant. The only result which seriously bucks the trend is the deterioration in C–C errors.

14.6.2 Between-category errors

Considering our main question again, of the seventeen categories of error studied, four show a significant trend towards improvement. Of the others, two show no significant change (N–J and J–R), and eight show a non-significant change towards improvement. There are two categories (C–R and I–T) which show a non-significant deterioration. Most interestingly, however, there is one category, N–R, which shows a significant deterioration in ability as the weeks progress.

The N–R distinction is populated with such words as *back*, *behind* and *past*. It is difficult to make a general rule which would help the student here. Some words are more frequently nouns than adverbs,[6] while others are more commonly adverbs than nouns.[7] It is conceivable that the students could receive more help from the system by being warned in advance that this was a difficult distinction. They could also be told that checking the frequency lexicon more often may help with this distinction. But what this finding could show is that the introduction of a clearer distinction between noun and adverb is a point at which traditional teaching methods and automated teaching may be able to join forces.

In reality the number of errors caused by the N–R distinction is low. Even so, we have also identified areas here in which the students are not developing significantly. Also there are a variety of distinctions with relatively high populations of errors, such as N–N. It may be that by augmenting automated tuition with human tuition specifically geare towards distinctions which the automated tuition does not seem to cope with well, we could achieve an even greater across the board improvement in grammatical competence amongst students of grammar.

14.7 Conclusion

In this chapter we have presented an application of annotated corpora in computer-aided learning. As a result of the exploitation of annotated corpora, an effective grammar teaching program has been developed and is in use at Lancaster and other universities. In addition to this, we have been able to form insights into the errors students make. These may allow us to gear traditional teaching towards being a supplement to automated teaching, especially in areas where students seem to benefit less from automated tuition. The end result of using annotated corpora in the automated teaching of grammar will mean better grammarians faster. As the next step, Cytor introduces the teaching of constituent structure analysis. Further levels of analysis, and teaching of other languages than English, can be added to the same basic instruction model. The package can be made more congenial to the learner by permitting various types of texts, including those collected by the learner, to be accessed and analysed.

Notes

1. On the other hand, the accuracy rate of manually-corrected grammatical annotation is likely to be high enough to be acceptable (see Section 17.3).
2. Note that annotation here may include partial annotation tasks, such as identifying the opening and/or closing of noun phrases in a sentence or a sentence-sequence.
3. With an additional two hours to cover the initial lecture and final experiment described later.
4. Systematic experiments on the efficacy of the program for the teaching of constituent structure analysis are under development.
5. The CLAWS2 tagset is similar to the C7 tagset; see Appendix III.
6. For example, in the core corpus of the BNC, *part* occurs as a singular common noun 389 times, a general adverb 4 times, and the infinitive form of a verb 7 times.
7. For example, in the core corpus of the BNC *still* is a general adverb 693 times, an adjective 21 times and a noun only twice.

15

The Exploitation of Multilingual Annotated Corpora for Term Extraction

TONY McENERY, JEAN-MARC LANGÉ, MICHAEL OAKES and JEAN VÉRONIS

Annotated multilingual corpora are allowing new approaches to known problems to be adopted. In this chapter, we will concentrate upon how annotated multilingual corpora can be used to generate terminology lists, for use in automated and human-aided translation.

15.1 Annotated Multilingual Corpora

Under initiatives from the European Commission, annotated multilingual corpora are becoming available to the language engineering community. Projects which Lancaster has participated in, such as ET-10/63,[1] MULTEXT[2] and CRATER[3] have been funded by the Commission to produce multilingual text collections, annotated for part-of-speech and translation equivalence. To take one project as an example, the CRATER project, which in its late stages entered into a collaboration with MULTEXT,[4] produced three TEI (Text Encoding Initiative) conformant corpora of approximately one million tokens each. Each corpus represented the same set of texts in one of three languages (English, French and Spanish); hence these corpora were **parallel corpora**. The texts in question were telecommunications manuals from the International Telecommunications Union (ITU).

In each corpus, each token was associated with a lemma and a part-of-speech code. Available in a separate file was a description of which sentences in the French corpus were translations of which sentences in the English corpus, and in a similar file which English sentences were translations of which Spanish sentences.

The production of such annotated multilingual resources is of undoubted use in a variety of applications within **machine translation**, such as example-based machine translation (Nagao 1984), statistically-based machine translation (Brown *et al.* 1990) and **terminology**

extraction. It is on the last of these applications that this paper will concentrate, in describing the use of multilingual corpora in projects ET10/63 and CRATER.

15.2 Terminology Extraction

The establishment of reliable technical terminology lists in translation has become an ever more important goal in translation studies and machine translation research, as the information revolution of the late twentieth century has progressed. As new concepts and devices need to be described, especially in technical texts, the creation of highly specific terms, which need precise translation, has grown. Schutz (1994) gives an account of the growing importance of terminology extraction in translation, and in machine translation in particular. To quote Schutz (1994: 3): 'Terminological data processing has grown largely in response to the information explosion which led to ... many new concepts ... and to strong interest in effective international communication.'

The creation and maintenance of reliable terminology databases is of necessity an ongoing process. This need is exacerbated by the need to translate technical texts in growing numbers in increasingly integrated international associations, such as the European Community. It is obvious, therefore, that the generation and maintenance of reliable terminology lists is of primary importance for translators, be those translators humans or machines. It is similarly obvious that it would be desirable for the process of the creation and maintenance of terminology lists to be automated.

Corpora are one source of data for automated terminology extraction. Realising this allows us to understand why corpora such as the CRATER and MULTEXT[5] corpora have been constructed. They almost certainly represent unbalanced and impoverished examples of the languages they instantiate, but when we consider the relationship between terms and technical texts, it is obvious that the type of corpus which would be of help here is precisely the type of corpus developed by CRATER and MULTEXT – consisting of technical texts which could be of use in deriving domain specific terminology. When we are looking at highly domain-specific features such as technical terminology, the narrow and specialized focus of the corpus texts is not merely appropriate but vital. Figuratively speaking, by looking across from one text to another, it is hoped that terms can be isolated and aligned between the languages of the parallel corpus.

Having established that specialized multilingual corpora may be of potential interest for terminology extraction, it is now appropriate to consider how this has been attempted.

15.3 The Gaussier/Langé/Daille Work on ET10/63

On project ET10/63, Gaussier, Langé and Meunier (1992) and Daille (1995) attempted to derive a bilingual terminology list automatically from two parallel corpora of technical texts. The corpora in question were earlier part-of-speech tagged and lemmatized versions of the French and English ITU corpora, later developed further in CRATER.

Using the part-of-speech annotation, a series of language-specific candidate term extraction automata were constructed, as described by Daille (1995). The automata generate a list of supposed candidate term sequences from their associated language. They do this by exploiting the part-of-speech tagging of the corpus, to identify morphosyntactic sequences associated with typical term constructions, such as compound nouns in English (e.g. $noun_1$-$noun_2$) and French (e.g. $noun_2$-*de*-$noun_1$).

The list generated by the automata is of lemmatized noun compounds, with content words only being represented, and associated function words deleted. So, for instance, the French compound *antenne de reception* is rendered as *antenne reception*. The term extraction automata generate a variety of constructions, not all of which are terms. To restrict the search solely to sequences of words which are closely associated, Daille (1995) adopted a strategy of using a co-occurrence statistic to measure the affinity between lexemes occurring in terms. The idea is that the elements of a compound technical term will have a higher association measure than will nouns which simply tend to co-occur. Daille assessed a variety of co-occurrence measures for term identification, and found that the **Cubic Association Ratio** (MI^3) measure (derived from **mutual information**)[6] sifted good terms from the candidate term list most effectively. This sifting process proceeded by testing the candidate terms produced by the finite-state automata for affinity, using this measure. Only those with a sufficiently large positive affinity were retained and were assumed to be good terms.

The techniques developed by Daille were taken up by Gaussier *et al.* who added two important refinements. They first tried to align terms not on the basis of whole corpus texts, but by restricting their search for translated terms to aligned sentences. This proved to improve significantly the results achieved by Daille. The other innovation that Gaussier *et al.* introduced was a best-match criterion to deal with conflicting potential translations of terms. The effects of Gaussier *et al.*'s amendment of the Daille technique were impressive. Gaussier and Langé (1994) reported that using this technique, they achieved an accuracy (precision) rate of 80 per cent over the top 500 candidate terms extracted. This accuracy level descended rapidly, however, the further down the list of candidate terms one went.

By the time one had reached the 1,200th pattern, the accuracy rate had dropped to around 6 per cent.

The work of the ET10/63 team was significant in that it demonstrated clearly how multilingual corpora could be used to tackle the question of automated terminology extraction. Where it is less effective, however, is in producing accurate terminology lists which require no human input. The work of ET10/63 was furthered in studies by the CRATER project, a key goal being to increase the precision of term extraction, even if this were at the expense of recall.[7]

15.4 The Use of Cognates for Terminology Extraction

Cognates have been proposed by various researchers, such as Simard *et al.* (1992), Johansson *et al.* (1993) and McEnery and Oakes (1995) as a means by which an improved measure of sentence alignment may be calculated. In the present context, by 'cognates' is meant simply mutually-translating terms which resemble one another orthographically, whether or not they are cognate in the sense of reflecting a common linguistic origin.

McEnery and Oakes (1995) developed a means of automatically extracting cognates from parallel texts in different languages, where there exists a common alphabet and a degree of lexical borrowing between the languages, using **approximate string matching techniques** (ASMTS). The most effective of these techniques was judged to be **Dice's similarity coefficient** (Dice 1945), originally developed to provide a means of comparing biological specimens. When applied to words, however, the technique counts how many letter pairs two strings have in common. The number they have in common is then expressed as a percentage of the total number of letter pairs that the strings possess. The higher this percentage, the more similar the two words are deemed to be. The calculation for the similarity coefficient, S, is $S = (2 * a) / (b + c)$, where a is the number of letter pairs in the (total number of shared letter pairs between) two given strings, b is the number of letter pairs in the first given string, and c is the number of letter pairs in the second given string. To give some examples, the pair *spectator/espectador* registers the respectable score of 70.6 per cent similarity when assessed by this method, while the pair *transmission/transmisión* receives an even higher score, 95.2 per cent.[8] Dice's similarity coefficient was used as a new means of assessing termhood on the CRATER project (McEnery and Oakes 1996).

15.5 Multiword Unit Alignment Using ASMT Derived Cognates

Work at Lancaster has now moved on to discovering whether ASMTs can be used to extract multiword cognates, assumed to be terms, from texts. Often it is not sufficient simply to identify cognates and to hope that concatenation of cognates will show up as multiword cognates in a text. These multiword units often tend to be a mixture of function and content words, with the function words tending not to be potential cognates, and the content words being potential cognates. In order to extract multiword units using ASMTs three experiments have been undertaken. These are described in this section.

The first experiment was undertaken using a naïve window matching algorithm, which took all *n*-sized sequences from one text, and compared them to all potential *m*-sized regions in an aligned region of the parallel text. Dice's similarity coefficient was computed on the windows. Dice was used not simply because McEnery and Oakes (1995) found it to be the best measure for single word cognate extraction, but because it is not sensitive to the order in which the words appear within a multiword comparison.[9] The windows with the highest Dice score are deemed to be the best translations.

To return to a discussion of the windowing algorithm, the *m*-sized sequence from the aligned region of text which gained the highest score against the *n*-sized region of text from the other language was recorded as the best potential cognate for that sequence.[10] Note that this algorithm is naïve in the extreme. No attempt is made to optimize the set of window fits between the two aligned regions, and no check is kept of whether a window in the second language is matched more than once. So obvious improvements to the algorithm are possible. Yet in spite of this naïveté, the results yielded by the program attempting four different window fits between English and Spanish, as shown in Figure 15.1, are quite remarkable. In the above figure, four window fits are attempted:

1. Two English words to two Spanish words (2×2)
2. Two English words to three Spanish words (2×3)[11]
3. Three English words to two Spanish words (3×2)
4. Three English words to three Spanish words (3×3)

All fits have similar characteristics, with two notable exceptions. The first exception is that although the experiment was run across a tenth of the Spanish-English parallel corpus (100,000 words of each language), no examples of a 90–100 per cent match was found for the 3×2 pattern

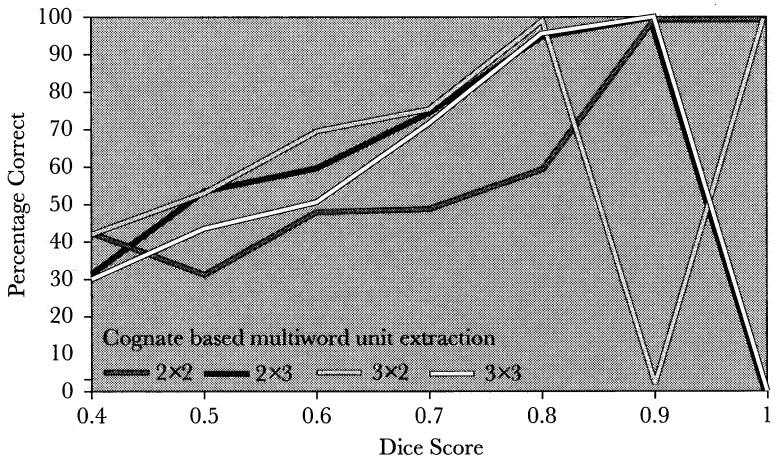


Figure 15.1 Dice's similarity coefficient calculated on multiword units between English and Spanish

match. Secondly, the 2×2 pattern is notably worse at the 80–90 per cent level than the other three window fits. With these exceptions noted, the pattern match was quite productive, and quite accurate – the higher the similarity score, the more likely it is that any proposed mutual translation for a multiword sequence based on similarity is correct. At the 80–89 per cent level, three of the window fits produce quite accurate results:

- 2×3 = 94.9 per cent
- 3×2 = 96.04 per cent
- 3×3 = 96.36 per cent

At 90 per cent and above, all of the patterns are perfectly accurate, with the exception of 3×2 and 2×3 at the 100 per cent level (no possible examples, as noted) and 3×2 at the 90–100 per cent level (no examples). All in all, this gives us a total of 3142 windows matched in a 1,000,000 word corpus, with 3033 reliable window matches and 109 bad alignments across the corpus if we accept 2×3, 3×2, 3×3 at the 80 per cent and above level, and 2×2 at the 90 per cent and above level[12] – an overall accuracy of 96.5 per cent.

The naïve windowing algorithm described worked quite well, aligning sequences such as *en caso de* and *in case of* purely and simply on the basis of similarity. There was no control of the linguistic features aligned, however, hence the alignment itself was of a whole range of cognate constructions. In the second and third experiments, we tried to extract specific

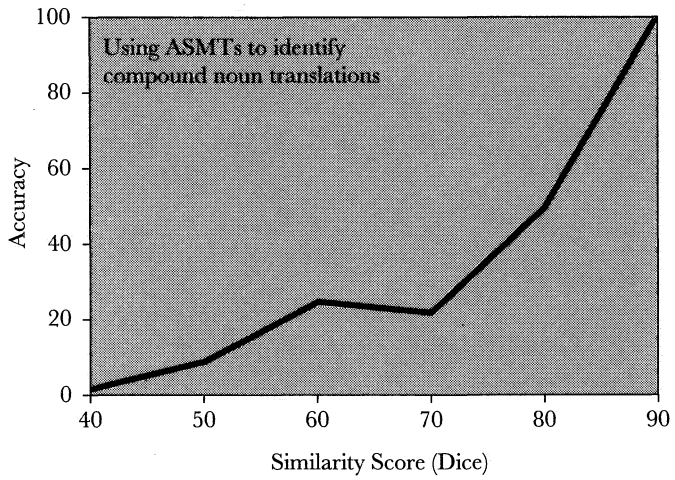


Figure 15.2 Using Dice's similarity coefficient to measure the cognateness of English and Spanish compound nouns

multiword units – compound nouns – and align them between the languages. For this the fact that the CRATER corpora are part-of-speech annotated and lemmatized[13] was of use. A focus on compound nouns was decided upon, as it has been noted, for example by Gaussier (1995), that compound nouns are an area where similarity and termhood seem to coincide. To test out Gaussier's speculation, finite state automata were developed which describe a range of typical compound noun constructions in Spanish, and separate ones developed (based on the one used by Daille 1995) to describe a series of typical compound noun constructions in English. In the first experiment, we replicated Daille's technique for candidate term list extraction and then used similarity with a best-match criterion applied as the sieve which decided which were good terms and which bad.

As shown in Figure 15.2, although the technique is reliable for high Dice similarity scores, the performance of the technique at lower levels of similarity is a cause for concern. Although one could imagine that restricting searches to aligned sentences could, once again, improve the performance, at least part of the problem, when the output was examined, was that the term extraction automata are not entirely reliable. Sequences which fit the pattern, but which are not noun compounds are extracted by the automata, for example, the sequence *caso de utilización*, in the sentence *En caso de utilización errónea, el equipo puede sufrir daños irreperables*,[14] would appear to be a noun compound to any automata looking for the

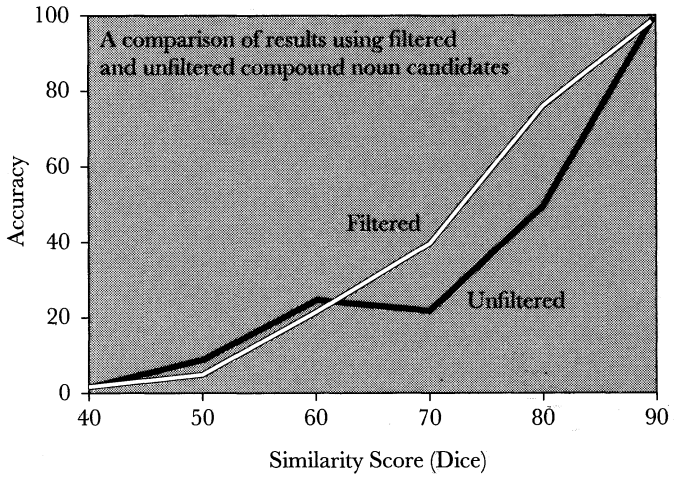


Figure 15.3 Using Dice's similarity coefficient to measure the cognateness of English and Spanish compound nouns

pattern *noun-de-noun* – but it is clearly not a compound noun in this case. Consequently, for our third experiment, we passed two sieves across the candidate term list. In a first pass, we used the Cubic Association Ratio to extract terms within which the lexemes had a positive association. In the second, we used Dice's similarity metric to determine which of these candidate terms were genuine terms. The result, as shown in Figure 15.3, was a marked improvement (both results are shown on the graph for purposes of comparison).

These results are more impressive, and work is progressing at Lancaster to incorporate alignment data into this comparison. But even so it is becoming increasingly obvious that compound nouns are susceptible to at least some measure of alignment using ASMTs. It would seem that Gaussier's (1995) observation is correct, therefore, and the potential impact of these findings is obvious. If similarity can be used as a useful metric for identifying the translations of noun compounds, then work which depends upon such identification, such as the term extraction work of Gaussier *et al.* (1992), Daille (1995) and Gaussier (1995) can benefit from such a technique.

Before leaving this discussion, however, it would be useful to consider how many good terms such a technique could retrieve from an annotated multilingual corpus. In Table 15.1 below, the number of terms extracted from a 10,000 word sample of the CRATER corpus are presented, and estimates of how the results achieved would scale up to a 1,000,000 word

corpus are presented, giving results for both English/Spanish and French/Spanish.

Table 15.1 Results of applying an ASMT as a termhood criterion

English/Spanish Comparison

Dice score	Number in 10,000 token corpus	Estimated numbers in 1,000,000 word corpus
.70– .79	12	1083
.80– .89	0	0
.90–1.00	3	271

Spanish/French Comparison

Dice score	Number in 10,000 token corpus	Estimated numbers in 1,000,000 word corpus
.70– .79	141	12,720
.80– .89	33	2,976
.90–1.00	3	271

In a 3×10,000 word section of the corpus (10,000 words each of parallel English, French and Spanish text) 202 English pattern types were identified yielding 318 patterns with 202 tokens with a positive affinity, 868 French pattern types yielding 1516 pattern tokens with 856 tokens with a positive affinity, and 524 Spanish pattern types yielding 1106 pattern tokens with 510 tokens with a positive affinity.

It is clear from the Table 15.1 that the result presented could lead to the retrieval of significant numbers of good terms. Where the technique gains its greatest advantage over the methods used on ET10/63 is that with the similarity score we have a measure which corresponds directly to accuracy – the user can decide which terms to examine on the basis of the similarity score. If one looked at terms with a score of 90 per cent or over, one would retrieve few terms, but they would all most likely be accurate. Lowering the threshold is a clear trade-off of precision against recall.

15.6 Conclusion

In this chapter we have tried to illustrate how similarity can be used to create a measure of multiword unit alignment between two texts. The experiments described hold the promise of translation unit alignment above the level of the word. Where, as in the experiments identifying

compound nouns, this leads to the chance of accurate recognition of a subset of a linguistic feature which is of interest, for example, to terminologists, then it is easy to see that annotated multilingual corpora are potentially of great importance in computational linguistics.

This said, however, there are limitations in the work presented here. With the exception of the 3×3 windowing method, the terms being extracted are generally two-part terms. Many terms are longer than 2 words (even after the application of the term extraction automata). Consequently, work needs to progress beyond that presented here before a general solution to the problem of automated terminology extraction can be achieved. Nevertheless, the results presented here, which would have been impossible without multilingual annotated corpora, represent good progress towards the final goal of automated terminology extraction.

Notes

1. Funded under the Eurotra programme of the European Union, 1992–93. Lead partner: IBM Paris.
2. Funded under the LRE programme of the European Union, 1994–96. Lead partner: Université de Aix-en-Provence.
3. Funded under the MLAP programme of the European Union, contract no. 93/20, 1994–95. Other partners: IBM Paris, C2V Paris, Universidad Autonóma de Madrid.
4. Additional funding from the MULTEXT project allowed the CRATER corpora to be converted into a TEI-conformant format.
5. MULTEXT has produced a series of 200,000 word parallel corpora based upon the Official Journal of the European Community. Each corpus is part-of-speech annotated.
6. $MI^3 = log_2(a^3/(a+b)(a+c))$, where a is the number of times two tested sequences co-occur, b is the number of times where we have the first element but not the second, and c is where we have the second element but not the first.
7. On **precision** and **recall**, see Chapter 7, n. 1.
8. Currently in all work at Lancaster accented characters are stripped from words and replaced by non-accented forms prior to any comparison being computed.
9. To give an example using the truncation, Dice and dynamic programming ASMTS, comparing *transmisión simultánea* to *simultaneous transmission*, would lead to a bad truncation score of 0, and a very bad dynamic programming score of 44 per cent. As Dice looks only at letter pairs, the sequences receive a very respectable 80 per cent from the Dice technique, correctly identifying them as candidate translations.
10. Such 'best match' criteria have been used to good effect in the processing of parallel corpora. Gaussier *et al.* (1992) used a best match criterion to improve the accuracy of their term-extraction technique, while McEnery and Oakes

(1996) used a best-match criterion to lower the effective threshold for identifying cognates using Dice's similarity coefficient to 70 per cent and above.

11. Note that perfect matches are, by definition, impossible when comparing three words to two and two words to three.
12. These figures are estimates based upon a detailed analysis of a 70,000 word English and 70,000 word Spanish aligned parallel corpus text. The texts were randomly selected from the corpus.
13. See McEnery *et al.* (1994) for a description of the annotated corpus.
14. 'In the case of erroneous use, the equipment may suffer irreparable damage.'

16

Towards Cross-Linguistic Standards or Guidelines for the Annotation of Corpora

PETER KAHREL, RUTHANNA BARNETT
and GEOFFREY LEECH

16.1 Introduction

The production of an annotated corpus is without doubt an expensive task, in terms of both time and effort, and therefore the reusability and shareability of such a resource is of great importance. Standardization of annotation practices can ensure that an annotated corpus can be used to its greatest potential. For an annotated corpus, standardization can be seen as important on two levels:

1. Standard encoding of corpora and annotations
2. Standard annotation of corpora

This first level has been addressed on a world-wide basis by the Text Encoding Initiative (TEI), using recommendations for the use of SGML for mark-up in corpora (see Sperberg-McQueen and Burnard 1994: Chapter 26).[1] The second level, which is being addressed in Europe by the EAGLES initiative (Expert Advisory Group for Language Engineering Standards), is the one with which we will be concerned in this chapter. As EAGLES represents the world's first major attempt at cross-linguistic annotation ground rules, in this chapter we will look in some detail at this initiative, and discuss the various problems encountered, and the possible solutions proposed.

We will first explain in more detail why standards are necessary (Section 16.2). In Section 16.3 we discuss a number of problems in connection with formulating standards and how these problems can be dealt with. Section 16.4 illustrates the EAGLES proposals that have been made recently for adopting standards. An important but often overlooked aspect of any annotation scheme is its documentation; this is taken up in the final Section 16.5.

The issues discussed in this chapter may apply to any level or type of

annotation (syntactic, semantic, morphosyntactic, phonological, pragmatic, etc.). However, the most widely applied annotations to date are morphosyntactic (tagging individual words; see Chapter 2) and, slightly less commonly, syntactic annotation (marking constituent structure and syntactic relations such as Subject and Object; see Chapter 3). Since these types of annotation produce similar and interrelated problems, we will concentrate mainly on these two levels in order to illustrate the issues involved in any attempt at standardization.

16.2 Why are Standards Considered Necessary?

There are a number of reasons why standards are helpful and, in many cases, necessary (see also Section 1.3):

1. Although a great deal of annotation work, both morphosyntactic and syntactic, has been done on English, many projects have been undertaking (or at least starting) work on other languages. Standardization of annotation practices will ensure to a degree that text corpora annotated by different groups in different countries are comparable, which is vital for research done, for example, on aligning parallel corpora of different languages (see Section 15.1).
2. Annotation work on only one language has been carried out by various research teams in various countries. Without standards, none of this work is easily comparable, and much unnecessary extra work would be required to allow further research on the same corpus.
3. Annotating a large corpus can be an extremely expensive activity, and in the current situation tools developed by one group cannot (or can hardly) be re-used by other groups. Setting annotation standards means that tools developed for the annotation of one corpus have a greater chance of being interchangeable and reusable, thus saving time, effort, and funds.
4. On a completely different level, standardization could also facilitate the exploitation of corpus research. Annotated corpora may be used for a variety of applications: many industries or research groups would benefit from the use of a corpus, but would not want to formulate their own annotation scheme from scratch. A standard scheme can act as an 'off-the-shelf product' which they can (relatively) easily implement.

16.3 Problems with Standardization

There are a number of problems associated with standardization of

annotation practices, which are reflected in the use of the term 'guidelines' alongside 'standards' in the title of this chapter.

1. *Relevance of standards to existing and parallel research* Any standards to be produced must take account of work previously carried out. The standards should be sufficiently flexible so that any already existing annotated corpus that has proved useful for its intended purpose will conform to the standards with little effort. The standards should also be compatible with parallel areas of research, e.g. lexicon building. Lexicons are not covered in detail in this book. However, since the structure of sentences is determined to a large extent by the lexical properties of the words that make up the sentences, the codings in the lexicon should be compatible with both the morphosyntactic and the syntactic annotation schemes. After all, it should be possible to annotate a corpus morphosyntactically for a particular task, but it should also be possible to use the tagged corpus later for other tasks such as syntactic annotation and lexicon enrichment.
2. *Acceptability of standards* Standardization of annotation seems to presuppose that there is agreement about the linguistic analysis of the corpus. But this is hardly the case. In the case of morphosyntactic annotation, it is virtually impossible to lay down rules for absolute consistency in the application of tags to text (see Section 17.1). Ideally, an annotation scheme should be so precise that when two annotators apply that scheme to a corpus, both annotations will be the same. But there are many fuzzy boundaries. For example, in English it is not clear whether one should analyse *gold* in *a gold watch* as a noun or an adjective. In syntactic annotation, not only do we have to determine which labels to apply to segments of the text, but the segments themselves have to be chosen from among many possibilities. The way these segments relate to one another also has to be determined. Fortunately, there is considerable consensus about some of the syntactic segments which have to be recognized in syntactic annotation – e.g. noun phrases and prepositional phrases. On the other hand, there is less consensus about how other syntactic segments should be defined, as illustrated by the following anecdote (Sampson 1995: 4). During the annual conference of the Association of Computational Linguistics in 1991, NLP researchers from nine institutions were asked to specify the bracketing of a number of example sentences. One of the examples was:

   > He said this constituted a [very serious] misuse [of the [Criminal Court] processes].

   The brackets here represent the only constituents the nine researchers could agree on: viz.: the adjective phrase *very serious*, the prepositional

phrase *of the Criminal Court processes* and the nominal constituent *Criminal Court.*

While standards must be explicit and usable, they cannot be too stringent or limiting. As shown above, there may be disagreement about the definitions or applicability of particular kinds of linguistic analysis. At the same time, while much of the work in this area is still at an early stage, a scheme chosen for annotation may have a marked effect on the success of the automation of this annotation. With syntactic annotation this can easily be illustrated – many syntactically annotated corpora are used as a training tool for automatic parsers. Since no completely successful parser has as yet been developed, the imposition of any particular scheme for syntactic annotation could be detrimental to future research.

3. *Task dependence of corpora* Strict standards pose a problem in an unrelated way as well. Annotation schemes may be produced for a wide variety of uses. An annotated corpus may be intended for use purely in linguistic research (e.g. studying variation in language use across genres, or across time; studying the frequency of particular vocabulary, or structures); or for natural language processing (as a testbed for an automatic parser; as example text for example-based translation, etc.). However, since the production of corpora is expensive in terms of time and effort, the most desirable corpus would be one that is suited to both theoretical and applied ends of the research spectrum. This is not so easy in practice – the aims of these two approaches could be very different, and this would be reflected in the corpus produced and the annotation applied to it. If a corpus is to be processed automatically, certain phenomena which may be problematic for automatic annotation may be left out for reasons of practical expediency, although these phenomena may be more interesting from a linguistic point of view. With reference more specifically to syntactic annotation, from an NLP perspective, the most important (and difficult) task may be the simple grouping together of certain parts of sentences into constituents, while from the linguist's perspective, this is the simplest (and mainly intuitive) task.
4. *Relevance of standards to a wide range of languages* Because much of the work on annotated corpora has been carried out on English, projects now underway on other languages may tend to be unduly influenced by that previous work. Cross-linguistic standards should be flexible enough to comprehend a wide range of languages. In the case of EAGLES, for the moment the aim is to move towards standardization in the treatment of European languages (initially those of the European Community), but optimally the emerging standards should be applicable to as many other languages or language families as possible.

For all these reasons standardization in the commonly-understood sense of 'seeking uniformity' is too confining in the current state of research: setting such standards would seriously constrain annotation practices without leaving any room for development. Furthermore, setting rigid standards does not acknowledge that annotated corpora can be used for different uses and, indeed, prohibits task dependent annotation.

So the dilemma the research community is faced with is: standards of annotation would be a good thing, but they are very difficult to implement in practice. Recently, in spite of the problems associated with standardization, EAGLES produced two documents on the standardization of morphosyntactic and syntactic annotation.[2] While recognizing all the pitfalls, these documents take the form of provisional guidelines (essentially, a tentative move towards standards) and sets of recommendations. They leave enough scope for researchers to vary.[3]

Now let us take a closer look at how the EAGLES recommendations have been formulated.

16.4 EAGLES Guidelines for Annotation

EAGLES has so far undertaken to propose sets of provisional guidelines for the morphosyntactic and syntactic annotation of corpora. To counter the danger of overrigidity mentioned above, three levels of constraint on annotation practices have been suggested. These three levels, **obligatory**, **recommended**, and **optional** annotations, are naturally different for morphosyntactic and syntactic annotation, but in both types of annotation the three levels are distinguished. We will discuss and illustrate these three levels in separate sections. The guidelines proposed for morphosyntax are discussed in Section 16.4.1; and for syntactic annotation, in Section 16.4.2.

16.4.1 Morphosyntactic annotation (or part-of-speech tagging)

The three levels of constraint proposed for morphosyntactic annotation are the following (Leech and Wilson 1994: 8):

1. **Obligatory attributes or values.**[4] These are characteristics that have to be included in any POS tagset. They include the major parts of speech, such as Noun and Verb.
2. **Recommended attributes or values**. These are widely recognized grammatical categories which occur in conventional grammatical

descriptions, such as Person, Number, Gender, Case and Tense, as well as major subcatgories such as 'Common' and 'Proper' for Nouns.

3. **Optional extensions to the list of attributes or values**. This category is subdivided into two:
   (a) Generic attributes or values. These are not usually encoded, but may be included by anyone tagging a corpus for specific purposes. For example, it may be desirable for some purposes to mark semantic classes such as temporal nouns, manner adverbs, place names, etc.
   (b) Language-specific attributes or values. These may be important characteristics of particular languages (e.g. honorifics in Japanese and other East Asian languages; cases in Finnish), and indeed might be recommended by someone annotating texts in those languages.

Below we illustrate a number of the levels.

Obligatory attributes/values

Only the major word categories, or parts of speech, are assigned to the obligatory level. These are the following (note that it is the categories, not the labels, that are obligatory):

1. N Noun
2. V Verb
3. AJ Adjective
4. PD Pronoun/Determiner
5. AT Article
6. AV Adverb
7. AP Adposition
8. C Conjunction
9. NU Numeral
10. I Interjection
11. U Unique/unassigned
12. R Residual
13. PU Punctuation

Most of these are familiar, and need no comment. The Adposition category subsumes both prepositions and postpositions. (Prepositions are, of course, dominant in the wider-known European languages; but arguably postpositions can be exemplified in the *'s* genitive morpheme and the temporal particle *ago* in English.) The Unique value (U) is applied to categories with a unique or very small membership, such as the infinitive marker (*to* in English, *zu* in German) and the existential particle (*there [is/are]*) in English, *er* in Dutch.

The residual value (R) is assigned to classes of word token which lie outside the range of 'canonical' grammatical classes, although they do occur quite commonly in many texts. For example, foreign words, or mathematical formulae. It can be argued that these are on the fringes of the grammar or lexicon of a language; nevertheless, they need to be

tagged – for automated corpus analysis no part of the text can be ignored.

Punctuation marks (PU) are (perhaps surprisingly) treated as a part of morphosyntactic annotation, as it is very common for punctuation marks to be tagged and to be treated as equivalent to words for the purpose of automatic tagging and corpus parsing.

Recommended attributes/values

Of the recommended attributes/values, we illustrate just one (Nouns):

Nouns
(i) Type 1. Common 2. Proper
(ii) Gender 1. Masculine 2. Feminine 3. Neuter
(iii) Number 1. Singular 2. Plural
(iv) Case 1. Nominative 2. Genitive 3. Dative 4. Accusative 5. Vocative

Here, Type, Gender, Number and Case are attributes; what follows the arabic numerals are values. For specific languages, both the attributes and the values can be easily extended as the need arises. For example, some languages have not only Singular and Plural, but also Dual number. Similarly for Case, many languages have fewer or more than the five cases listed under (iv). The number of attributes can be extended as well, to handle languages with different alignment systems than nominative-accusative, e.g. ergative languages like Basque.

16.4.2 Syntactic annotation ('treebanks')

Guidelines for syntactic annotation need to take account of a number of 'flexibility' issues, as discussed in Section 16.3 (3–4). Among the reasons for flexibility are: (a) annotated corpora can be used for a wide variety of uses (we called this 'task dependence' above); (b) since annotation practices are still developing, it would be inadvisable to impose a straitjacket on such an immature research area. This caveat is even more true of syntactic annotation than of morphosyntactic annotation. It follows that EAGLES should not propose one standard in this area, but, rather, a set of preliminary recommendations. To handle the first problem, the EAGLES documentation specifies a number of different layers of annotation. Roughly in order of increasing complexity or abstraction, these layers are as follows (cf. Table 3.1, p. 49):

(a) Bracketing of segments;
(b) Labelling of segments;
(c) Marking of dependency relations (see Section 3.3.5);
(d) Indicating functional labels, such as Subject and Object;

(e) Marking subclassification of syntactic segments;
(f) Deep or 'logical' information;
(g) Information about the rank of a syntactic unit (e.g. Clause, Phrase, Word);
(h) Special syntactic characteristics of spoken language.

By allowing these different layers of annotation, without making any of them obligatory, the guidelines meet the requirement that a corpus can be annotated appropriately for a specific purpose. We briefly illustrate some of these layers below.

(a) **Bracketing of segments** consists in the delimitation by some annotative device (for our purposes, square brackets) of sentence segments (normally hierarchically organized) which are recognized as having a syntactic integrity (e.g. sentences, clauses, phrases, words). For example:

[[He] [walked [into [the garden]]]]

(b) **Labelling of segments** amounts to specifying the formal category of the non-terminal syntactic units or constituents identified by bracketing, such as Noun Phrase, Verb Phrase, Relative Clause. Thus adding labels to the above string might yield the following labelled analysis:

[S [NP He NP] [VP walked [PP into [NP the garden NP] PP] VP] S]

(c) **Marking subclassification of syntactic segments**. This means assigning attribute values to constituents such as clauses or phrases, e.g. marking a Noun Phrase as singular, or a Verb Phrase as past tense. A feature-based syntax has been modelled by the Text Encoding Initiative (Sperberg-McQueen and Burnard 1994), in which syntactic information may be represented by means of attribute-value pairs. If necessary, this kind of information can be added in a compact way by adding various subscripts to syntactic segments. Figure 16.1 is an example from the SUSANNE Corpus (showing only part of a sentence), with, in the right hand column, a singular (proper) noun phrase (Nns), and past tense verb phrase (Vd).

A01:0010b	AT	The	the	[O[S[Nns:s.
A01:0010c	NP1s	Fulton	Fulton	[Nns.
A01:0010d	NNL1cb	County	county	.Nns]
A01:0010e	JJ	Grand	grand	.
A01:0010f	NN1c	Jury	jury	.Nns:s]
A01:0010g	VVDv	said	say	[Vd.Vd]

Figure 16.1 Marking subclassification in SUSANNE

(f) **Marking logical (or deep structure) relations of various kinds.** This includes a variety of syntactic phenomena, such as co-referentiality (for example in control structures), cross-reference (or substitution), ellipsis, traces, and syntactic discontinuity. Such information is found, for example, in the SUSANNE Corpus and in the second phase of the Penn Treebank (see Sections 3.3.2 and 3.3.4).

(h) **Information about spoken language non-fluency phenomena.** Spoken language corpora show a range of phenomena that do not normally occur in written language corpora, such as blends, false starts, reiterations, and filled pauses. In syntactic annotation, it has to be decided whether to include such phenomena in a parse tree, and if so, how. There is now increasing interest in this layer of annotation. For example, the British National Corpus contains a small syntactically-annotated subcorpus with inclusion of non-fluency phenomena in the skeleton parsing of spoken data (Eyes 1996). Sampson (personal communication) is now beginning a new project extending the SUSANNE Corpus to spoken data, as discussed in Sampson (1995: Ch. 6). There is also a proposal to include an analysis of such phenomena in the parsing of the British component of the International Corpus of English (Aarts 1992, Greenbaum 1992).

A second issue mentioned at the beginning of this section is that it is not advisable to set standards in a research field that is still developing. The EAGLES guidelines cater for this in two unrelated ways. First, no standard is set for the formalization of syntactic structures or relations. It is recognized that there are two main methods of representing syntactic relations in terms of tree-like structures: phrase structure and dependency structure (see Section 3.3.5). However, there are no good reasons for accepting one of these as a standard and rejecting the other. Second, as with the guidelines for morphosyntactic annotation, the information types given in the guidelines for syntactic annotation are specified on three levels of constraint:

1. Obligatory annotations
2. Recommended annotations
3. Optional annotations.

Obligatory

Because of the variable nature of syntactic annotation, and the many combinatorial possibilities, it is suggested that no part of the syntactic annotation be treated as obligatory. The first layer (a) in the 'hierarchy' of annotation, bracketing, could be seen to be a possibly obligatory level, and indeed for a constituent structure analysis, it would be. However, as we have seen, there are dependency-based schemes that do not actually

group together the words making up constituents (e.g. ENGCG – see Section 3.3.5), and these must still undoubtedly be regarded as a useful form of syntactic annotation.

Recommended

On the recommended level, certain non-terminal categories are proposed as annotations within a phrase structure model. They comprise the widely recognized major constituents:

Sentence/Clause	Adjective Phrase
Noun Phrase	Adverb Phrase
Verb Phrase	Prepositional Phrase

as well as coordination phenomena. Although these non-terminal categories are widely recognized, it is not easy to agree on precisely how they are instantiated in texts. The documentation accompanying a corpus should therefore give a clear account of how these constituents are defined, with sufficient attention to problem cases.

Optional

On the optional level, such annotation types as the following are suggested, being commonly useful in parsing and in providing syntactic information in the lexicon:

- sentence subcategorization: different sentence types (declarative, imperative, interrogative)
- syntactic clause subcategorization: clauses annotated as to their formal or functional characteristics (nominal, adverbial, relative, etc.)
- syntactic phrase subcategorization: further subcategorization of phrases to include values such as Person, Number, Case, Tense, Voice and Aspect.
- grammatical function: inclusion of syntactic functions such as Subject, Object, Indirect Object.
- semantic phrase subcategorization: specification of semantic functions of constituents, such as Locative, Temporal adverbials.
- deep/logical information: annotation of various phenomena indicated in (f) above.

16.5 Documentation: a standard after all

There is one area which deserves obligatory standards, namely the documentation of the annotation scheme (see Section 1.3). Without adequate

documentation provided by its originators, an annotated corpus can be extremely difficult for other users to apply to their own research tasks. Decisions taken in the development of an annotation scheme, as well as in its application, should be well documented in order to ensure that future users will be able to apply the scheme in a manner consistent with that of the originators of the scheme, and which will then be consistent in the new application. It would be unrealistic to expect optimal documentation practices; but at least the documentation should include some reference to each of the following classes of information:

(a) *What level or layers of annotation have been undertaken?* The documentation should include information as to what particular phenomena are marked in the annotation scheme.

(b) *What is the set of annotation devices used (e.g. brackets, labels) and what are the meanings of these devices?* Each symbol should be described, defined and illustrated with one or more examples.

(c) *What are the conventions for the application of the annotation devices to texts?* An annotation scheme[5] (i.e. a tagging scheme or parsing scheme) is more than (a) and (b) above. It includes the set of guidelines or conventions whereby the annotation symbols are to be applied to text sentences, such that (ideally) two different annotators, implementing the scheme manually to the same sentence, would agree on the analysis to be applied (see further Sections 1.3, 2.5, 3.3.4). To increase its coverage, an annotation scheme may include reference to a lexicon, to a grammar or to a reference corpus of annotated sentences.

(d) *What is the measurable quality of the annotation?* Answers to this should include: (i) to what extent the corpus has been manually checked; (ii) accuracy rate; (iii) consistency rate. These different measures of quality of annotation will depend mainly on how the corpus is annotated. An automatic annotation will require figures of accuracy – often given in terms of a percentage success rate, or in terms of recall and/or precision (see Chapter 7, n. 1; also Voutilainen 1995). A recall of less than 100 per cent indicates that some correct readings have been discarded, while a precision of less than 100 per cent indicates that superfluous readings remain in the output in the form of system ambiguities.

(e) *How detailed/shallow is the analysis?* To a certain extent, the specificity of the analysis may be shown by the levels/layers of annotation that have been applied. However, more detailed documentation may be necessary in order to make clear the granularity or level of detail to which an annotation is undertaken – for example some aspects of a deep or logical grammar may be included in an annotation, while

others are not marked (e.g. marking of discontinuity, but no marking of 'traces').

(f) *To what extent and in what respects has disambiguation (of machine-generated ambiguities) been carried out?* During the annotation of a corpus, ambiguous structures may be left in the mark-up (see Section 9.3). Resolution of problematic ambiguities should be documented, as should any ambiguities that are left in the corpus.

(g) *To what extent and in what respects is the annotation at any particular level or layer incomplete?* At any particular level of annotation, certain markings may be ignored by the annotation scheme, for ease of automated annotation, or because of the intended purpose of the resource. Information of this sort should also be included in the documentation.

Standardization is difficult and, especially in the case of syntactic annotation, controversial to the extent that it will be impossible to formulate one agreed 'consensus' standard. Therefore EAGLES proposes tentative standards on different levels to accommodate different theoretical approaches to language.[6]

Notes

1. Examples of TEI conformant encoding of corpus annotation are given in Section 2.4 and Chapter 3, n. 8.
2. Morphosyntactic guidelines are presented in Leech and Wilson (1994), and syntactic guidelines in Leech, Barnett and Kahrel (1995).
3. It should be emphasized that the guidelines are preliminary and subject to later modification. A critique of the syntactic annotation guidelines (Leech, Barnett and Kahrel 1995) is provided by Atwell (forthcoming).
4. The use of 'attribute' and 'value' is illustrated as follows: *Feminine* is a value of the attribute *Gender*; *Singular* is a value of the attribute *Number*.
5. Also termed a 'grammatical representation' by Voutilainen (1994).
6. The provisional EAGLES recommendations on morphosyntactic and syntactic annotation of corpora were the result of teamwork. We gratefully acknowledge the contributions of the following committee members: Gerardo Arrarte, Nicoletta Calzolari, Paula Guerreiro, Jean-Marc Langé, Monica Monachini, Simonetta Montemagni, Anne Schiller, Hans van Halteren, and Atro Voutilainen.

17

Consistency and Accuracy in Correcting Automatically Tagged Data

JOHN PAUL BAKER

17.1 Introduction

One of the reasons for determining standards for annotation (as discussed in the last chapter) is that it facilitates the evaluation of annotation practice. Criteria for evaluating the achievement or quality of one annotation project against another have not so far been well developed. In grammatical tagging, for example, it has been considered sufficient to calculate the success rate of an automatic tagger by hand-checking the output against an implicit standard of 'what is correct' (see Sections 1.3 and 2.5). A single percentage accuracy figure, derived from the number of correctly tagged word tokens divided by the total number of word tokens (ignoring punctuation tags) has been considered sufficient. This, it is true, is a useful indicator of tagging quality, but only in so far as it rests on a clear and acceptable definition of 'what is correct'. As we have seen, there is no such clear definition – although the detailed specification of an annotation scheme (or tagging scheme) can progressively approximate to it.

In this situation, it has been easy to criticize the value of using human post-editors to correct automatically tagged data. Sinclair (1992) argues that human checking is futile because ambiguities exist in language, and it is preferable to emphasize rather than camouflage the indeterminacy of the state of our grammars at present. A similar point of view, in some ways, has been put forward by Church,[1] who maintains there is a residue of up to 5 per cent of words about which human judges cannot agree what is correct, simply because of the element of disorderliness and indeterminacy in human language. Further, it can be hypothesized that using human post-editors decreases the internal consistency of the tagged data. A computer will not deviate from its programming, whereas humans, due to inattention, boredom or overfamiliarity, make slips. Thus a single human post-editor might spot a mistake made by an automatic tagger 99 times out of 100, but would fail to notice every error, thus introducing a

level of inconsistency into the data. Also, several humans working on the same corpus might disagree over the tagging protocol of certain words, creating another level of inconsistency, whereas although an automatically tagged corpus might contain a larger proportion of errors, at least those errors would remain consistent throughout the corpus.[2] Apart from slips, humans could make errors because a proportion of the words in the corpus could be assigned more than one tag and still be classed as 'correct'. Even though ambiguous words might present a problem for a computer, the introduction of human (subjective) post-editors might prove to be another obstacle to accurate and consistent tagging.

Against this sceptical position, a more positive view (which is adopted here) is that ultimately it is the human being's mental interpretation that enables us to evaluate the quality of annotation. Automatic tagging or parsing which bore no relation to this mental interpretation would be valueless. However, because the human analyst is susceptible to error and inconsistency, the mental interpretation of what is correct has to be sharpened and made explicit through the specification of an annotation scheme. With the help of such an explicit scheme, it is hypothesized that human post-editors can achieve a high degree of accuracy and consistency, even though the ideal of 100 per cent may not be achievable.

This chapter reports on an experiment which was designed to test the above hypothesis, and to discover to what extent using human post-editors to check automatically tagged corpora would introduce inconsistencies in the data. Four experienced post-editors were given sentences of written and spoken data from the BNC, which had previously been tagged by CLAWS, and asked to remove errors from the output. Mean rater accuracy was found to be higher than the accuracy of CLAWS output (99.11 per cent to 96.95 per cent), while overall consistency between post-editors was 98.8 per cent. At the time of this experiment, the tagger used for the BNC, CLAWS, had an accuracy of 96–97 per cent when using the C7[3] tagset, so human post-editors were employed to check and correct the rogue 3–4 per cent of errors in the sampler corpus.

Recently, other experiments carried out upon human post-editors have attempted to address the issue of single and inter-rater consistency. Marcus *et al.* (1993) carried out a consistency experiment using four annotators, each to correct a 16,000 word sample of the Brown Corpus, half of which had been automatically tagged using PARTS (Church 1988), while the other half was tagged manually using the Penn Treebank tagset (containing 36 POS tags and 12 punctuation/currency tags). They found a mean inter-rater agreement of 96.5 per cent, and mean accuracy of 96.6 per cent for the correcting task as opposed to PARTS accuracy of 91.4 per cent. Voutilainen and Jarvinen (1995) carried out a similar experiment

with two human post-editors and the English Constraint Grammar Parser ENGCG (Karlsson *et al.* 1995) upon a 6,000-word sample of text. The ENGCG used 139 POS tags, and initial consistency between the two humans was reported at above 99 per cent. However, the former experiment used a tagset with fewer distinctions, possibly making it easier to be consistent, while the latter experiment only had two raters, reducing the possibility of inconsistency. Baker (1995) carried out a preliminary inter-rater consistency experiment that attempted to combine the strengths of the work of Marcus *et al.* and Voutilainen and Jarvinen, by using the large C7 tagset (136 POS tags) and a larger number of post-editors (9). Subjects post-edited a 2,183-word sample that had been automatically tagged by CLAWS. Mean rater accuracy was found to be higher than the accuracy of CLAWS output (96.9 per cent to 93.3 per cent), while overall consistency between post-editors was 96 per cent. However, the sample contained many problematic and difficult-to-tag word sequences (for both CLAWS and human post-editors), as the intention of the experiment had been to highlight areas where guidelines could be tightened, as well as to determine inter-rater consistency. As a result of that experiment, changes were made to CLAWS, and to the tagging scheme, resulting in a more standardized system of both human and automatic tagging.

It was therefore decided to repeat Baker's experiment, using a sample of a similar size, consisting of subsamples taken at random from the BNC, rather than a concocted sample that had been hand-picked to represent a worst case scenario. A random sample would thus give a more accurate reading not only of the accuracy of CLAWS, but of the post-editing abilities of the subjects.

17.2 Method

(a) *Subjects* Four members of the UCREL research team participated in the experiment. Although there were differences between subjects in amount and recency of post-editing experience, all subjects had worked on the BNCTE (British National Corpus Tagging Enhancement project – see Chapter 9) for the previous eighteen months, and as a result all were familiar with the C7 tagset. The subjects were a subset of those who had also participated in the earlier inter-rater consistency experiment (Baker 1995): the other five who did not participate had not been involved in work with the C7 tagset during the previous twelve months, and were therefore not aware of the changes in tagging protocol that had occurred during that time.

(b) *Materials* Subjects were provided with a 42-page booklet containing 118 sentences (1,970 words in total, once all punctuation had been subtracted) which had been tagged by CLAWS. The first 74 sentences (1,576 words in total) were chosen at random from the written part of the BNC. The sentences were chosen from many different BNC texts and included a political news story, a science report, an article about weddings, a letter about antiques and an extract from society pages from a magazine. The last 44 sentences (394 words in total) were taken from the spoken data section of the BNC and consisted of parts of transcripts of a business meeting and a conversation at home, which represented the context-governed and demographic sections of the spoken BNC respectively. The spoken section of the booklet was much smaller than the written section, in order to reflect to an extent the larger proportion of written data in the BNC (90 per cent written compared to 10 per cent spoken).

(c) *Procedure* Subjects were each given a booklet and requested to check the decisions that CLAWS had made, noting the cases where CLAWS had made an incorrect choice, and retagging the word correctly. Subjects were advised to maintain the levels of diligence and speed they were accustomed to for any other post-editing task, and not to confer with anybody either during or after participation in the experiment. However, they were encouraged to refer to the official in-house tagging scheme (*Post-Editor's Guide to CLAWS C7 Tagging*[4]), the on-line lexicons and idiom lists which were used by CLAWS as part of the tagging process, or any personal notes they had made in the past in order to make their task easier. A copy of the C7 tagset was made available to anybody who needed it. (Subjects were given the opportunity to remain anonymous by sealing their booklets in an envelope, although nobody took this option.)

(d) *Results* All of the subjects completed the whole booklet. To calculate **inter-rater consistency** the following formula was used: for a single word, consistency was equal to the number of pairs of agreements divided by the number of pairs of possible agreements. The mean consistency was then calculated by summing consistency for all words and dividing by the total number of words. Thus, if everyone agreed on a single word, consistency for that word would be 1, whereas if three subjects agreed with each other and one disagreed, consistency would be 3 (the number of agreements) divided by 6 (the number of possible agreements), equaling 0.5.

Accuracy was determined by conformity to tagging guidelines, as verifed by the experimenter. In no case was there any ambiguity

about the identification of the correct tag for a particular word. Also, accuracy and consistency were determined for both written and spoken data separately, as well as together. The results are shown in Table 17.1.

Table 17.1 Percentage of correctly annotated words for all subjects (H1–H4 represent the four human raters)

	Written	Spoken	All
CLAWS	97.21	95.93	96.95
H1	99.30	98.50	**99.14**
H2	98.92	99.24	**98.98**
H3	98.73	98.22	**98.63**
H4	99.62	100	**99.69**
H (mean accuracy)	99.14	98.99	**99.11**
H(mean consistency)	98.9	98.3	**98.8**

Errors were examined to distinguish slips from genuine mistakes. For the purposes of this experiment, slips are identified as being cases where CLAWS originally made the mistake, and subsequently the error went unnoticed by the post-editor. Genuine errors, however were classified as cases where the post-editor actually changed a tag, resulting in the word being incorrectly post-edited. Table 17.2 shows the breakdown of errors for each post-editor.

Table 17.2 Errors categorized according to type for all subjects

	Number of errors	Number of slips	Number of genuine errors
H1	17	14	3
H2	20	19	1
H3	27	25	2
H4	6	4	2
Proportion of all errors	70	62 (88.57%)	8 (11.43%)

Frequencies were also calculated for errors made according to the POS category to which they were assigned. As most categories are divided into various subsets (e.g. there are nineteen types of pronoun tag, all beginning with the letter 'P'), error types have been collapsed into simpler categories whenever possible: the sets *adverb*, *det*, *noun*, *pronoun*, *adj* and *verb* consist of all adverbs, all determiners, all nouns, all pronouns, all adjectives and all

verbs respectively. In this way it is possible for one class of noun (e.g. NN1) to be incorrectly tagged as another class of noun (e.g. NP1). Table 17.3 shows frequencies of all such errors.

Table 17.3 Errors made by human post-editors

	Incorrect tag group									
Tag that should have been assigned	adverb (R*)	APPGE	det (D*)	Prep (II)	adj (J*)	MC1	noun (N*)	Intj (UH)	Vb-ing (VVG)	**Total**
adverb (R*)	2		2	3	1					**8**
conjunction (C*)			4	6						**10**
postdet.sg (DA1)	1									**1**
exist. *There* (EX)	2									**2**
preposition (II)	5						1	2		**8**
adjective (JJ)							5			**5**
noun (N*)					2		22		2	**26**
pronoun (P*)		6				2				**8**
verb (V*)					1		1			**2**
Total	**10**	**6**	**6**	**9**	**4**	**2**	**29**	**2**	**2**	**70**

* is used as a wild card, so that e.g. N* signals all tags beginning with N.

17.3 Discussion of Results

All subjects, and CLAWS, performed as had been expected. The normal accuracy for CLAWS is 96–97 per cent, and on the random sample of data, CLAWS achieved a good 96.95 per cent accuracy. Mean post-editor accuracy improved on this score by more than 2 per cent (to 99.11 per cent) and the four post-editors agreed on decisions 98.8 per cent of the time (still almost 2 per cent higher than the accuracy of CLAWS output alone). Comparisons between the written and spoken data sections are not so dramatic. Human performance was overall 0.15 per cent better on written data, while CLAWS was 1.28 per cent better on written data. This slight improvement in performance on written data could be because spoken data tends to be 'ungrammatical'; sentences can be fragmentary or left unfinished, and the transcripts themselves can give the post-editor an inaccurate impression. Also, CLAWS had originally been designed for and trained on written data.

The majority (88.57 per cent) of human errors resulted from slips, the

post-editor failing to spot a mistake that had originally been made by CLAWS. The 'genuine errors' that were caused by post-editors changing a correct tag to an incorrect one amounted to only eight cases.

For the experimental data, as Table 17.3 shows, if a word was tagged incorrectly 41.4 per cent of the time it would be (incorrectly) assigned a noun tag by human post-editors. Also, the percentage of errors made where the correct tag should have been noun tag was 37.1 per cent. Although there are a high number of nouns in the data set (29.08 per cent), they are by no means the only high-frequency set of tags: e.g., verbs make up 19.69 per cent of the sample, but incorrectly-assigned verb tags only accounted for 2.8 per cent of all of the errors. The high proportion of noun-related errors is possibly due to a degree of ambiguity between the different types of noun tags. One problem in the text was with the tagging of phrases such as *Colonel, the Hon Sir Gordon Palmer* where the word *Colonel* was automatically tagged as NN1 (singular common noun) whereas most of the post-editors tagged it as NNB (preceding singular noun of style or title). This phrase occurred several times in one part of the text, and there was inter-rater inconsistency even within subjects. It was determined (in the light of the guidelines) that in this unusual construction the NNB tag was appropriate. Of the 29 incorrect noun assignments, 22 (75.86 per cent) were due to the wrong noun tag being assigned. However, noun-noun errors of this type are not as grave as other errors, e.g. noun-verb or noun-adjective.

When the results of this experiment are compared to the results of the earlier inter-rater consistency experiment carried out at Lancaster University, improvements can be seen in every field. Inter-rater consistency improved (from 96 per cent to 98.8 per cent), mean accuracy of human post-editors improved (from 96.9 per cent to 99.11 per cent) and the performance of CLAWS also improved (from 95.3 per cent to 96.95 per cent). The enhanced performance of both human post-editors and the automatic tagger could be due partly to the fact that the data set in this experiment was taken at random, as compared to one that had been chosen in order to highlight problematic areas. Other reasons for improvement were (a) that since the previous experiment, the tagging scheme guidelines had been tightened in order to eliminate many cases of ambiguity, and also (b) that the resources of CLAWS itself had been improved, with e.g. a much larger and more discriminating lexicon, and many additions to the idiom list, which handles difficult tag sequences.

The results are encouraging: although it cannot be expected that any post-edited corpus will be 100 per cent free of errors, a fair experiment produced results that gave a mean post-editor accuracy of over 99 per cent and an inter-rater consistency of almost 99 per cent. Therefore the

argument that it is pointless to let one (or more) humans post-edit a piece of automatically tagged data because it would introduce a high degree of inconsistency into the data appears to have been refuted by this experiment. Not only did all of the human post-editors improve upon the 'raw' computer data: but their collective consistency was still higher than that of the computer alone. It remains to be seen whether other levels of corpus annotation are capable of yielding similar results.

Notes

1. During a lecture course held at the University of Stockholm, August 1996.
2. This argument does not hold where software is undergoing development, nor does it apply to statistical annotation software, where the results of automatic annotation are less easy to predict, and where the same software may produce inconsistent results in apparently similar contexts.
3. The C7 tagset is listed in Appendix III.
4. This document can be consulted at http://www.comp.lancs.ac.uk/ucrel/claws7pe.html.

Appendix I
Sources for Further Information

World Wide Web and Email Addresses

1. Corpora

The British National Corpus
http://info.ox.ac.uk/bnc/
email: natcorp@oucs.ox.ac.uk

Corpora available from ICAME (Brown, LOB, SEC, Helsinki, etc.)
http://www.hd.uib.no/corpora.html

The CRATER corpus and browser
http://www.comp.lancs.ac.uk/linguistics/crater/corpus.html

The ICE Corpus
http://www.ucl.ac.uk/english-usage/ice.htm

The Lampeter Corpus of Early Modern English Tracts
http://www.tu-chemnitz.de/~ehe/emode.htm

The Oxford Text Archive
http://sable.ox.ac.uk/ota/
ftp://ota.ox.ac.uk/pub/ota/public

The SEU (Survey of English Usage) Corpus
http://www.ucl.ac.uk/english-usage/survcorp.htm

The SUSANNE Corpus and Analytic Scheme
http://www.cogs.susx.ac.uk/users/geoffs/RSue.html

2. Software

AMALGAM (this also has an email/WWW tagging service)
http://www.scs.leeds.ac.uk/amalgam/amalgam/amalgsoft.html

Eric Brill's Home Page (this includes links to his tagger software)
http://www.cs.jhu.edu/~brill/

The COSMAS Concordancer
http://www.ids-mannheim.de/ldv/cosmas/intro.html

The IMS Corpus Toolbox
http://www.ims.uni-stuttgart.de/~oli/CorpusToolbox/

LEXA Software
http://www.hd.uib.no/lexainf.html

SARA (this includes a useful link to other SGML software)
http://info.ox.ac.uk/bnc/sara.html

WordSmith Tools
http://www1.oup.co.uk/oup/elt/software/wsmith?

The Xerox ('Cutting') Tagger
ftp://parcftp.xerox.com/pub/tagger/

3. Other Useful Sites

UCREL (University Centre for Computer Corpus Research on Language, Lancaster) http://www.comp.lancs.ac.uk/ucrel/
This contains a page of links to other sites of interest to corpus linguists. For information about availability of UCREL software described in this book,
email: ucrel@lancaster.ac.uk

AMALGAM Project Home Page at Leeds
http://www.scs.leeds.ac.uk/amalgam/amalgam/amalghome.htm

CobuildDirect Birmingham
http://titania.cobuild.collins.co.uk/direct_info.html

EAGLES
http://www.ilc.pi.cnr.it/eagles/home.html

EAGLES Spoken Language Working Group
http://coral.lili.uni-bielefeld.de/~gibbon/EAGLES/

ELRA (European Language Resources Association)
http://www.icp.grenet.fr/ELRA/home.html

ICAME (International Computer Archive of Modern and Medieval English) Home Page http://www.hd.uib.no/icame.html

Institut für angewandte Kommunikations- und Sprachforschung e.V., Bonn (a German corpus, the writings of Kant, plus some software)
http://cl1.ikp.uni-bonn.de/IKS/

Institut für deutsche Sprache
http://www.ids-mannheim.de/

Lingsoft (including the Helsinki Constraint Grammar Tagger/Parser ENGCG)
http://www.lingsoft.fi/

Linguistic Data Consortium (LDC)
http://www.ldc.upenn.edu/

Multext (Multilingual Text Tools and Corpora)
http://www.lpl.univ-aix.fr/projects/multext/

Natural Language Software Registry
http://cl-www.dfki.uni-sb.de/cl/registry/

Norwegian Computing Centre for the Humanities, Bergen
http://www.hd.uib.no/e-index.html

Research Unit for Multilingual Language Technology, Helsinki
http://www.ling.helsinki.fi/research/rumlat.html

Society for Conceptual and Content Analysis by Computer (SCACC)
email: schmidt@opie.bgsu.edu

TOSCA (Tools for Syntactic Corpus Analysis, Nijmegen)
http://lands.let.kun.nl/research/tosca/togen.html

Appendix II
Glossary of Abbreviations and Acronyms

AGFL	Affix Grammar over Finite Lattices
AMAZON	Automatische Zinsontleding ('Automatic sentence analysis'; Dutch)
AP	Associated Press
APHB	American Printing House for the Blind corpus
ASCII	American Standard Code for Information Interchange
ASMT	Approximate String Matching Techniques
ATR	Advanced Telecommunications Research
BNC	British National Corpus
BNCTE	British National Corpus Tagging Enhancement project
CCPP	Computer Corpus Pilot Project
CFPSG	Context-Free Phrase Structure Grammar
CGEL	*Comprehensive Grammar of the English Language*
CHILDES	Child Language Data Exchange System
CLAWS	Constituent Likelihood Automatic Word-tagging System
COLT	Corpus Of London Teenage English
CQL	Corpus Query Language
CRATER	Corpus Resources And Terminology ExtRaction
DAARC	Discourse Anaphora and Anaphor Resolution Colloquium
EAGLES	Expert Advisory Group on Language Engineering Standards
ENGCG	English Constraint Grammar
ENGTWOL	English Two-Level Morphological Analysis
EPSRC	Engineering and Physical Sciences Research Council
ESRC	Economic and Social Research Council
FTP	File Transfer Protocol
GCE	*A Grammar of Contemporary English*
GWB	Grammarian's Workbench
HMM	Hidden Markov Model
ICAME	International Computer Archive of Modern and Medieval English
ICE	International Corpus of English
ICECUP	ICE Corpus Utility Program
ICLE	International Corpus of Learner English

IHE	Innovation in Higher Education
ITU	International Telecommunications Union
KWIC	Key Word In Context
LDB	Linguistic DataBase
LDC	Linguistic Data Consortium
LDOCE	*Longman Dictionary Of Contemporary English*
LLC	London-Lund Corpus
LOB	Lancaster-Oslo/Bergen Corpus
MLT	Research Unit for Multilingual Language Technology (Helsinki)
MT	Machine Translation
NLP	Natural Language Processing
OTA	Oxford Text Archive
OUCS	Oxford University Computing Services
POS	Part Of Speech
POW	Polytechnic Of Wales Corpus
PS	Phrase Structure
RAM	Random Access Memory
SARA	SGML-Aware Retrieval Application
SASG	Syntactic Annotation SubGroup (of EAGLES)
SCACC	Society for Conceptual and Content Analysis by Computer
SEC	Spoken English Corpus
SFG	Systemic Functional Grammar
SGML	Standard Generalized Mark-up Language
SUSANNE	Surface and Underlying Structure Analysis of Natural English
TCWG	Text Corpus Working Group (of EAGLES)
TEI	Text Encoding Initiative
TOBI	Tones and Break Indices
TOSCA	Tools for Syntactic Corpus Analysis
UCREL	University Centre for Computer Corpus Research on Language (Lancaster)

Appendix III

Specimen Annotation Practices: The C7 and C5 Tagsets

C5 Tagset (words exemplifying categories are added in italics)

AJ0	adjective (unmarked) (e.g. *good*, *old*)
AJC	comparative adjective (e.g. *better*, *older*)
AJS	superlative adjective (e.g. *best*, *oldest*)
AT0	article (e.g. *the*, *a*, *an*)
AV0	adverb (unmarked) (e.g. *often*, *well*, *longer*, *furthest*)
AVP	adverb particle (e.g. *up*, *off*, *out*)
AVQ	wh-adverb (e.g. *when*, *how*, *why*)
CJC	coordinating conjunction (e.g. *and*, *or*)
CJS	subordinating conjunction (e.g. *although*, *when*)
CJT	the conjunction *that*
CRD	cardinal numeral (e.g. *3*, *fifty-five*, *6609*) (excluding *one*)
DPS	possessive determiner form (e.g. *your*, *their*)
DT0	general determiner (e.g. *these*, *some*)
DTQ	wh-determiner (e.g. *whose*, *which*)
EX0	existential *there*
ITJ	interjection or other isolate (e.g. *oh*, *yes*, *mhm*)
NN0	noun (neutral for number) (e.g. *aircraft*, *data*)
NN1	singular noun (e.g. *pencil*, *goose*)
NN2	plural noun (e.g. *pencils*, *geese*)
NP0	proper noun (e.g. *London*, *Michael*, *Mars*)
ORD	ordinal (e.g. *sixth*, *77th*, *last*)
PNI	indefinite pronoun (e.g. *none*, *everything*)
PNP	personal pronoun (e.g. *you*, *them*, *ours*)
PNQ	wh-pronoun (e.g. *who*, *whoever*)
PNX	reflexive pronoun (e.g. *itself*, *ourselves*)
POS	the possessive (or genitive morpheme) *'s* or *'*
PRF	the preposition *of*
PRP	preposition (except for *of*) (e.g. *for*, *above*, *to*)
PUL	punctuation – left bracket (i.e. (or [ )
PUN	punctuation – general mark (i.e. . ! , : ; - ? ...)
PUQ	punctuation – quotation mark (i.e. ' ' " ")

PUR	punctuation – right bracket (i.e.) or])
TO0	infinitive marker *to*
UNC	'unclassified' items which are not words of the English lexicon
VBB	the 'base forms' of the verb *be* (except the infinitive), i.e. *am*, *are*
VBD	past form of the verb *be*, i.e. *was*, *were*
VBG	*-ing* form of the verb *be*, i.e. *being*
VBI	infinitive of the verb *be*
VBN	past participle of the verb *be*, i.e. *been*
VBZ	*-s* form of the verb *be*, i.e. *is*, *'s*
VDB	base form of the verb *do* (except the infinitive)
VDD	past form of the verb *do*, i.e. *did*
VDG	*-ing* form of the verb *do*, i.e. *doing*
VDI	infinitive of the verb *do*
VDN	past participle of the verb *do*, i.e. *done*
VDZ	*-s* form of the verb *do*, i.e. *does*
VHB	base form of the verb *have* (except the infinitive), i.e. *have*
VHD	past tense form of the verb *have*, i.e. *had*, *'d*
VHG	-ing form of the verb *have*, i.e. *having*
VHI	infinitive of the verb *have*
VHN	past participle of the verb *have*, i.e. *had*
VHZ	*-s* form of the verb *have*, i.e. *has*, *'s*
VM0	modal auxiliary verb (e.g. *can*, *could*, *will*, *'ll*)
VVB	base form of lexical verb (except the infinitive) (e.g. *take*, *live*)
VVD	past tense form of lexical verb (e.g. *took*, *lived*)
VVG	*-ing* form of lexical verb (e.g. *taking*, *living*)
VVI	infinitive of lexical verb
VVN	past participle form of lex. verb (e.g. *taken*, *lived*)
VVZ	*-s* form of lexical verb (e.g. *takes*, *lives*)
XX0	the negative *not* or *n't*
ZZ0	alphabetical symbol (e.g. *A*, *B*, *c*, *d*)

C7 Tagset (words exemplifying categories are added in italics)[1]

APPGE	possessive pronoun, pre-nominal (*my*, *your*, *our*)
AT	article (unmarked) (e.g. *the*, *no*)
AT1	singular article (e.g. *a*, *an*, *every*)
BCL	before-clause marker (e.g. *in order [that]*)
CC	coordinating conjunction (*and*, *or*)
CCB	adversative coordinating conjunction (*but*)
CS	subordinating conjunction (e.g. *if*, *because*, *unless*)
CSA	*as* as a conjunction
CSN	*than* as a conjunction
CST	*that* as a conjunction
CSW	*whether* as a conjunction
DA	after-determiner (capable of pronominal function; unmarked) (e.g. *such*, *former*, *same*)

DA1	singular after-determiner (e.g. *little*, *much*)
DA2	plural after-determiner (e.g. *few*, *several*, *many*)
DAR	comparative after-determiner (*more*, *less*)
DAT	superlative after-determiner (*most*, *least*)
DB	before-determiner (capable of pronominal function; unmarked) (*all*, *half*)
DB2	plural before-determiner (capable of pronominal function) (*both*)
DD	determiner (unmarked) (capable of pronominal function) (e.g. *any*, *some*)
DD1	singular determiner (e.g. *this*, *that*, *another*)
DD2	plural determiner (*these*, *those*)
DDQ	*wh*-determiner (e.g. *which*, *what*)
DDQGE	*wh*-determiner, genitive (*whose*)
DDQV	*wh-ever* determiner (e.g. *whichever*, *whatever*)
EX	existential *there*
FO	formula
FU	unclassified word
FW	foreign word
GE	Germanic genitive marker – (' or '*s*)
IF	*for* as a preposition
II	general preposition (e.g. *in*, *by*, *at*)
IO	*of* as a preposition
IW	*with; without* as prepositions
JJ	general adjective
JJR	general comparative adjective (e.g. *older*, *better*, *bigger*)
JJT	general superlative adjective (e.g. *oldest*, *best*, *biggest*)
JK	catenative adjective (e.g. *able* in *be able to*; *willing* in *be willing to*)
MC	cardinal number, neutral for number (*two*, *three*...)
MC1	singular cardinal number (*one*)
MC2	plural cardinal number (*tens*, *twenties*)
MCMC	hyphenated number (e.g. *40–50*, *1770–1827*)
MD	ordinal number (e.g. *first*, *2nd*, *next*, *last*)
ND1	singular noun of direction (e.g. *north*, *southeast*)
NN	common noun, neutral for number (e.g. *sheep*, *cod*)
NN1	singular common noun (e.g. *book*, *girl*)
NN2	plural common noun (e.g. *books*, *girls*)
NNA	following noun of style or title, abbreviatory (e.g. *M.A.*)
NNB	preceding singular noun of style or title, abbreviatory (e.g. *Prof.*)
NNL1	singular locative noun (e.g. *street*, *Bay*)
NNL2	plural locative noun (e.g. *islands*, *roads*)
NNO	numeral noun, neutral for number (e.g. *dozen*, *thousand*)
NNO2	plural numeral noun (e.g. *hundreds*, *thousands*)
NNT1	singular temporal noun (e.g. *day*, *week*, *year*)
NNT2	plural temporal noun (e.g. *days*, *weeks*, *years*)
NNU	unit of measurement, neutral for number (e.g. *in.*, *cc.*)

NNU1	singular unit of measurement (e.g. *inch*, *centimetre*)
NNU2	plural unit of measurement (e.g. *inches*, *centimetres*)
NP	proper noun, neutral for number (e.g. *Indies*, *Andes*)
NP1	singular proper noun (e.g. *London*, *Jane*, *Frederick*)
NP2	plural proper noun (e.g. *Browns*, *Reagans*, *Koreas*)
NPD1	singular weekday noun (e.g. *Sunday*)
NPD2	plural weekday noun (e.g. *Sundays*)
NPM1	singular month noun (e.g. *October*)
NPM2	plural month noun (e.g. *Octobers*)
PN	indefinite pronoun, neutral for number (e.g. *none*)
PN1	singular indefinite pronoun (e.g. *one*, *everything*, *nobody*)
PNQO	oblique case *wh*-pronoun (*whom*)
PNQS	nominative case *wh*-pronoun (*who*)
PNQV	nominative case *wh-ever* pronoun (*whoever*)
PNX1	reflexive indefinite pronoun (*oneself*)
PPGE	nominal possessive personal pronoun (e.g. *mine*, *yours*)
PPH1	3rd person singular personal pronoun (*it*)
PPHO1	3rd person oblique case singular personal pronoun (e.g. *him*, *her*)
PPHO2	3rd person oblique case plural personal pronoun (*them*)
PPHS1	3rd person nominative singular personal pronoun (*he*, *she*)
PPHS2	3rd person nominative plural personal pronoun (*they*)
PPIO1	1st person oblique case singular personal pronoun (*me*)
PPIO2	1st person oblique case plural personal pronoun (*us*)
PPIS1	1st person nominative singular personal pronoun (*I*)
PPIS2	1st person nominative plural personal pronoun (*we*)
PPX1	singular reflexive personal pronoun (e.g. *yourself*, *itself*)
PPX2	plural reflexive personal pronoun (e.g. *yourselves*, *ourselves*)
PPY	2nd person personal pronoun (*you*)
RA	adverb, after nominal head (e.g. *else*, *galore*)
REX	adverb introducing appositional constructions (*namely*, *viz*, *eg.*)
RG	degree adverb (e.g. *very*, *so*, *too*)
RGQ	*wh-* degree adverb (*how*)
RGQV	*wh-ever* degree adverb (*however*)
RGR	comparative degree adverb (*more*, *less*)
RGT	superlative degree adverb (*most*, *least*)
RL	locative adverb (e.g. *alongside*, *forward*)
RP	prepositional adverb; particle (e.g. *in*, *up*, *about*)
RPK	prepositional adverb, catenative (e.g. *about* in *be about to*)
RR	general adverb (*soon*, *quickly*, *perhaps*)
RRQ	*wh-* general adverb (*where*, *when*, *why*, *how*)
RRQV	*wh-ever* general adverb (e.g. *wherever*, *whenever*)
RRR	comparative general adverb (e.g. *better*, *longer*)
RRT	superlative general adverb (e.g. *best*, *longest*)
RT	nominal adverb of time (e.g. *now*, *tomorrow*)
TO	infinitive marker (*to*)

UH	interjection (e.g. *oh, yes, um*)
VB0	finite base form of the verb BE (*be*)
VBDR	past tense *-re* form of BE (*were*)
VBDZ	past tense *-s* form of BE (*was*)
VBG	*-ing* form of BE (*being*)
VBI	infinitive *be*
VBM	first person singular present tense of BE (*am*)
VBN	past participle form of BE (*been*)
VBR	present tense *-re* form of BE (*are*)
VBZ	present tense *-s* form of BE (*is*)
VD0	finite base form of the verb DO (*do*)
VDD	past tense of DO (*did*)
VDG	*-ing* form of DO (*doing*)
VDI	infinitive *do*
VDN	past participle form of DO (*done*)
VDZ	*-s* form of DO (*does*)
VH0	finite base form of HAVE (*have*)
VHD	past tense of HAVE (*had*)
VHG	*-ing* form of HAVE (*having*)
VHI	infinitive *have*
VHN	past participle form of HAVE (*had*)
VHZ	*-s* form of HAVE (*has*)
VM	modal auxiliary (e.g. *can, will, would*)
VMK	modal catenative (*ought, used*)
VV0	base form of lexical verb (e.g. *give, work*)
VVD	past tense form of lexical verb (e.g. *gave, worked*)
VVG	*-ing* form of lexical verb (e.g. *giving, working*)
VVGK	*-ing* form in a catenative verb (e.g. *going* in *be going to*)
VVI	infinitive of lexical verb (e.g. [*to*] *give*, [*will*] *work*)
VVN	past participle form of lexical verb (e.g. *given, worked*)
VVNK	past participle of a catenative verb (e.g. *bound* in *be bound to*)
VVZ	*-s* form of lexical verb (e.g. *gives, works* etc.)
XX	*not, n't*
ZZ1	singular letter of the alphabet: *A, a, B*, etc.
ZZ2	plural letter of the alphabet: *As, b's*, etc.

Notes

1. Punctuation tags are omitted from this list. The twelve punctuation marks *! " ( ) , - . ... : ; ?* ---- are tagged as themselves.

Bibliography

Aarts, J. (1992) 'Comments on a New Corpus of English: ICE.' In Svartvik (1992), 180–3.

Aarts, J., de Haan, P., and Oostdijk, N. (1993) (eds) *English language corpora: Design, analysis and exploitation.* Amsterdam: Rodopi.

Altenberg, B. (1991) 'A bibliography of publications relating to English computer corpora.' In Johansson and Stenström (1991), 355–95.

Altenberg, B. (1995) ICAME Bibliography 3 (1990–4). Available on line from ICAME, Norwegian Computing Centre for the Humanities, Bergen.

Ariel, M. (1988) 'Referring and accessibility.' *Journal of Linguistics* 24, 65–87.

Aston, G. and Burnard, L. (1995) 'The BNC handbook. Exploring the British National Corpus with SARA' (preliminary draft). Oxford: Oxford University Computing Services.

Atwell, E. S. (forthc.) 'Comparative evaluation of grammatical annotation models.' *Proceedings of the Workshop on Industrial Parsing of Software Manuals.* Amsterdam: Rodopi.

Atwell, E. S. and Elliott, S. (1987) 'Dealing with ill-formed English text.' In Garside *et al.* (1987), 120–38.

Baker, J. P. (1995) *The evaluation of multiple post-editors: inter-rater consistency in correcting automatically tagged data.* UCREL Technical Papers 7.

Baldinger, K. (1975) *Dictionnaire onomasiologique de l'ancien Gascon.* Tübingen: Max Niemeyer.

Banfield, A. (1973) 'Narrative style and the grammar of direct and indirect speech.' *Foundations of Language* 10, 1–39

Banfield, A. (1982) *Unspeakable sentences.* London: Routledge.

Beattie, B. (1983) *Talk: an analysis of speech and non-verbal behaviour in conversation.* Milton Keynes: Open University Press.

Black, E. (1993) 'Statistically-based computer analysis of English.' In Black *et al.* (1993), 1–16.

Black, E. (1993a) 'Additional topics in statistical grammar development.' In Black *et al.* (1993) 144–70.

Black, E. (1993b) Parsing English by computer: the state of the art. Invited presentation, *Proceedings, International Workshop on Spoken Dialogue,* Waseda Univer-

sity, Tokyo. (Also appears in: *Proceedings, ATR International Workshop on Speech Translation.* ATR Interpreting Telecommunications Research Laboratories, Kyoto.)

Black, E., Garside, R., and Leech, G. (1993) (eds) *Statistically-driven computer grammars of English: The IBM/Lancaster approach.* Amsterdam: Rodopi.

Black, E., Jelinek, F., Lafferty, J., Magerman, D., Mercer, R., and Roukos, S. (1992) 'Towards history-based grammars: Using richer models for probabilistic parsing.' *Proceedings of the Workshop on Speech and Natural Language*, Defense Advance Research Projects Agency, U.S. Government.

Black, E., Eubank, S., Kashioka, H., Garside, R., Leech, G., and Magerman, D. (1996) 'Beyond Skeleton Parsing: Producing a Comprehensive Large-Scale General-English Treebank With Full Grammatical Analysis.' *Proceedings, COLING-96*, Copenhagen.

Blackwell, S. (1987) 'Syntax versus orthography: Problems in the automatic parsing of idioms.' In Garside *et al.* (1987), 110–19.

Boguraev, B. and Briscoe, E. J. (1989) (eds) *Computational lexicography for natural language processing.* London: Longman.

Botley, S. P. (1996) 'Comparing demonstrative features in three written English genres.' In Botley *et al.* (1996a).

Botley, S. P. and McEnery, A. M. (forthc.) (eds) *Corpus-based and computational approaches to discourse anaphora.* London: UCL Press.

Botley, S. P., Glass J., McEnery A. M., and Wilson A. (1996a) (eds) *Approaches to Discourse Anaphora: Proceedings of the DAARC96 Colloquium*, UCREL Technical Papers Special Issue, Volume 8, Lancaster University.

Botley, S. P., Glass J., McEnery A. M., and Wilson A. (1996b) (eds) *Proceedings of Teaching and Language Corpora 1996*, UCREL Technical Papers Special Issue, Volume 9, Lancaster University.

Brill, E. (1992) 'A simple rule-based part-of-speech tagger.' *Proceedings of the 3rd Conference on Applied Natural Language Processing.* Italy: Trento.

Brill, E. (1993) Automatic grammar induction and parsing free text: a transformation-based approach ACL (Association for Computational Linguistics).

Brill, E. (1994) Some advances in transformation-based part-of-speech tagging. In *Proceedings of the 12th National Conference on Artificial Intelligence* (AAAI-94).

Brill, E. (1996) 'Learning to parse with transformations.' In H. Bunt and M. Tomita (eds), *Recent advances in parsing technology*, Dordrecht: Kluwer Academic Press.

Briscoe, T. (1994) 'Prospects for the parsing of unrestricted text: Robust statistical parsing techniques.' In Oostdijk and de Haan (1994), 97–119.

Briscoe, T. and Carroll, J. (1993) 'Generalised probabilistic LR parsing of natural language (corpora) with unification-based grammars.' *Computational Linguistics* 19(1): 25–60.

Briscoe, T. and Carroll, J. (1996) 'A probabilistic LR parser of part-of-speech and punctuation labels.' In Thomas and Short (1996), 135–50.

Brodda, B. (1991) 'Doing corpus work with PC Beta; or, how to be your own computational linguist.' In Johansson and Stenström (1996), 259–82.

Brown, G. and Yule, G. (1983) *Discourse analysis*. Cambridge: Cambridge University Press.

Brown, P. F., Cocke, J., Della Pietra, S. A., Della Pietra, V. J., Jelinek, F., Lafferty, J. D., Mercer, R. L., and Roosin, P. S. (1990) 'A statistical approach to machine translation.' *Computational Linguistics* 16(2): 79–85

Burnage, G. and Dunlop, D. (1993) 'Encoding the British National Corpus.' In Aarts *et al.* (1993), 79–95.

Burnard, L. (1995) (ed.) *Users' reference guide for the British National Corpus version 1.0*. Oxford: Oxford University Computing Services.

Chanod, J.-P. and Tapanainen, P. (1995). Tagging French – comparing a statistical and a constraint-based method. *Proceedings of the* EACL*-95*, Dublin.

Christ, O. (1994) A modular and flexible architecture for an integrated corpus query system. *Proceedings of* COMPLEX*'94: 3rd Conference on Computational Lexicography and Text Research (Budapest, July 7–10 (1994))*. Budapest, Hungary, 23–32.

Church, K. (1988) 'A stochastic parts program and noun phrase parser for unrestricted text.' *Proceedings of the Second Conference on Applied Natural Language Processing.* (ACL) Austin, TX, 136–43.

Church, K. (1992) 'Current practice in part of speech tagging and suggestions for the future.' In Simmons (ed.), *Sbornik Praci: In Honour of Henry Kučera*. Michigan: Slavonic Studies.

Church, K. (1993) 'Char_align: a program for aligning texts at the character level.' *Proceedings of the Association of Computational Linguistics, (1993)*, 1–8.

Church, K. and Mercer, R. L. (1993) 'Introduction to a special issue on computational linguistics using large corpora.' *Computational Linguistics* 19 (1), 1–24

Church, K., Mercer, R. L., Gale, W., Hanks, P., and Hindle, D. (1990) 'Using statistics in lexical analysis.' Unpublished paper, Bell Laboratories and Oxford University Press.

Conte, M.-E. (1996) 'Facts, events, propositions in anaphoric encapsulation.' Paper presented at the IndiAna Workshop on Indirect Anaphora, Lancaster University, UK.

Cook, G. (1995) 'Theoretical issues: transcribing the untranscribable.' In Leech *et al.* (1995) 35–53.

Coulmas, F. (1986) *Direct and indirect speech*. Berlin: Mouton de Gruyter.

Crowdy, S. (1993) 'Spoken corpus design and transcription.' *Literary and Linguistic Computing*, 8(4), 259–65.

Crowdy, S. (1995) 'The BNC Spoken Corpus.' In Leech *et al.* (1995), 224–34.

Cunningham, H., Wilks, Y., and Gaizauskas, R. (1996) 'GATE – a General Architecture for Text Engineering.' *Proceedings of* COLING*–96*.

Cutting, D. and Pedersen, J. (1993) *The Xerox Part-of-Speech Tagger*. Xerox Palo Alto Research Center, Palo Alto, CA.

Cutting, D., Kupiec J., Pedersen, J., and Sibun, P. (1992) 'A practical part-of-speech tagger.' *Proceedings of the Third Conference on Applied Natural Language Processing, Trento.*

Dagan, I. and Itai, A. (1990) 'Automatic processing of large corpora for the resolu-

tion of anaphora references.' *Proceedings of the 13th International Conference on Computational Linguistics,* COLING *'90*, Helsinki.

Daille, B. (1995) *Combined approach for terminology extraction: lexical statistics and linguistic Filtering.* UCREL Technical Papers Number 5, Department of Linguistics, Lancaster University.

Day, A. C. (1992) *Roget's Thesaurus of the Bible.* London: Marshall Pickering.

DeRose, S. (1988) Grammatical category disambiguation by statistical optimization, *Computational Linguistics* 14 (1), 31–9.

DeRose, S. (1991) An analysis of probabilistic grammatical tagging methods. In Johansson and Stenström (1991), 9–14.

Derouault, A. M. and Merialdo, B. (1986) Natural language modelling for phoneme-to-text transcription. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* PAMI-8: 742–9.

Dice, L. R. (1945) Measures of the amount of ecologic association between species, *Ecology* 26, 297–302.

Dornseiff, F. (1933) *Der deutsche Wortschatz nach Sachgruppen.* Berlin: Walter de Gruyter.

Du Bois, J. W. and Schuetze-Coburn, S. (1993) Representing hierarchy: constituent structure for discourse databases. In J. A. Edwards and M. D. Lampert (eds) *Talking data: transcription and coding in discourse research.* Lawrence Erlbaum: Hillsdale, NJ 221–60.

Du Bois, J. W., Paolino, D., and Cumming, S. (1990) *Discourse transcription.* University of California, Santa Barbara: Linguistics Department.

Dunning, T. (1993) 'Accurate methods for the statistics of surprise and coincidence.' *Computational Linguistics* 19 (1), 61–74.

Ellegård, A. (1978) *The syntactic structure of English texts: A computer-based study of four kinds of text in the Brown University Corpus.* Göteborg: Gothenburg Studies in English 43.

Elliot, R., Hill, C. E., Stiles, W. B., Friedlander, M. L., Mahrer, A. R., and Margison, F. R. (1987) 'Primary therapist response modes: comparison of six rating systems', *Journal of Consulting and Clinical Psychology* 55.

Elworthy, D. (1995) 'Tagset design and inflected languages.' In *From texts to tags: issues in multilingual language analysis. Proceedings of the* ACL SIGDAT *Workshop*, Dublin, 1995.

Eubank, S. G., Kashioka, R., and Black, E. (1996) A new approach to treebank creation. In *Proceedings of the Second Annual Meeting of* NLP 265–8.

Eyes, E. J. (1996) The BNC Treebank: syntactic annotation of a corpus of modern British English, M.A. dissertation, Lancaster University

Eyes, E. J. and Leech, G. (1993) Syntactic annotation: Linguistic aspects of grammatical tagging and skeleton parsing. In Black *et al.* (1993) 36–61.

Fawcett, H. (1989) *A Text Searching System:* PAT3. *3, User's Guide.* Centre for the New Oxford English Dictionary, University of Waterloo.

Fawcett, R. and Perkins, M. (1980) *Child language transcripts* 6–12 (with a preface, in 4 volumes). Re-issued with light amendments (1981), Department of Behavioural and Communication Studies, Polytechnic of Wales.

Feldweg, H. (1995) Implementation and evaluation of a German HMM for POS disambiguation. In *Proceedings of the EACL SIGDAT workshop (1995)*, Dublin.

Fligelstone, S. (1991) A description of the conventions used in the Lancaster Anaphoric Treebank Scheme (2nd edition). UCREL Technical Report, Department of Linguistics, Lancaster University.

Fligelstone, S. (1992) Developing a scheme for annotating text to show anaphoric relations. In G. Leitner (ed.) *New directions in English language corpora: Methodology, results, software.* Berlin: Mouton de Gruyter, 153–70.

Fligelstone, S. (1993) 'Some reflections on the question of teaching, from a corpus linguistics perspective.' *ICAME Journal* 17, 97–109.

Fligelstone, S. (1995) 'Jaws: using lemmatisation rules and contextual disambiguation rules to enhance CLAWS output.' Project report, Linguistics Department, Lancaster University.

Fligelstone, S., Rayson, P., and Smith, N. (1996) Template analysis: bridging the gap between grammar and the lexicon. In Thomas and Short (1996), 181–207.

Francis, G. (1994) Labelling discourse: an aspect of nominal-group lexical cohesion. In R. Coulthard (1994) *Advances in written text analysis.* London: Routledge.

Francis, W. N. (1980) A tagged corpus – problems and prospects. In S. Greenbaum, G. Leech, and J. Svartvik (eds) *Studies in English linguistics for Randolph Quirk.* London: Longman, 192–209.

Francis, W. N. and Kučera, H. (1964) *Manual of information to accompany a Standard Sample of Present-day Edited American English, for use with digital computers.* Providence R.I.: Department of Linguistics, Brown University.

Francis, W. N. and Kučera, H. (1982) *Frequency analysis of English usage. Lexicon and grammar.* Boston: Houghton Mifflin.

Gaizauskas, R. and Humphries, K. (forthc.) Quantitative evaluation of coreference algorithms in an information extraction system. In Botley and McEnery (forthc.).

Gale, W. A., Church, K. W., and Yarowsky, D. (1992) 'One sense per discourse.' In *Proceedings of the 4th DARPA Speech and Natural Language Workshop*, 233–7.

Garside, R. (1993) The marking of cohesive relationships: tools for the construction of a large bank of anaphoric data, *ICAME Journal* 17, 5–27.

Garside, R. (1996) The robust tagging of unrestricted text: the BNC experience. In Thomas and Short (1996), 167–80.

Garside, R. and McEnery, A. (1993) Treebanking: The compilation of a corpus of skeleton parsed sentences. In Black *et al.* (1993) 17–35.

Garside, R. and Leech, F. (1985) 'A probabilistic parser.' In *Proceedings of the Second Conference of the European Chapter of the ACL* (Geneva, March 1985), 166–70.

Garside, R. Leech, G., and Sampson, G. (1987) (eds) *The computational analysis of English: A corpus-based approach.* London: Longman.

Garside, R., McEnery, A., and Rayson, P. (1993) Argument Frame Extraction and Term Clustering from an English-French Bilingual Aligned Corpus. ET10–63 Working Paper.

Garwick, A. W., Detzner, D., and Boss, P. (1994) Family perceptions of living with Alzheimer disease. *Family Process* 33: 3, 327–40.

Gaussier, E. (1995) Modèles statistiques et patrons morphosyntactiques pour l'extraction de lexiques bilingues, Ph.D. Thesis, Université de Paris VII, Jussieu.

Gaussier, E. and Langé, J.-M. (1994) Some methods for the extraction of bilingual terminology. International Conference on New Methods in Language Processing (NeMLaP), 14–16 Sept 1994. In Proceedings of the Conference, ed. Daniel Jones, UMIST, Manchester, 242–7

Gaussier, E. and Meunier, F. (1992) Towards bilingual terminology. *Proceedings of ALLC/ACH Conference*, Oxford: Oxford University Press, 121–24.

George, A. L. (1973) *Propaganda analysis: A study of inferences made from Nazi propaganda in World War II.* Westport, Conn: Greenwood Press.

Gottschalk, L. A. and Gleser, G. C. (1969) *The measurement of psychological states through content analysis of verbal behaviour.* Berkley, Cal.: University of California Press.

Granger, S. (1993) The International Corpus of Learner English. In Aarts *et al.* (1993), 57–69.

Greenbaum, S. (1992) A new corpus of English: ICE. In Svartvik (1992), 171–9.

Greene, B. B. and Rubin, G. M. (1971) *Automatic grammatical tagging of English.* Providence, R.I.: Department of Linguistics, Brown University.

Hall, P. A. V. and Dowling, G. R. (1980) Approximate string matching, *Computing Surveys*, 12: 4, 381–402.

Halliday, M. A. K. (1994) *An introduction to functional grammar*, London: Arnold.

Halliday, M. A. K. and Hasan R. (1976) *Cohesion in English*, London: Longman.

Hallig, R. and von Wartburg, W. (1952) *Begriffssystem als Grundlage für die Lexikographie: Versuch eines Ordnungsschemas.* Berlin: Akademie-Verlag.

Haslerud, V. and Stenström A.-B. (1995) The Bergen Corpus of London Teenager Language (COLT). In Leech *et al.* (1995) 235–42.

Heidebrecht, N. M. (1993) A lexicon of metal terminology in the Hebrew Scriptures with special reference to the excavations of the metal industry at Tel Dan, Israel. Ph.D. dissertation, University of California at Los Angeles.

Hickey, R. (1993a) Lexa: Corpus processing software. 3 vols. Volume 1: Lexical analysis. Volume 2: Database and corpus management. Volume 3: Utility library. Bergen: Norwegian Computing Centre for the Humanities.

Hickey, R. (1993b) Corpus data processing with Lexa, *ICAME Journal* 17, 73–95.

Hindle, D. (1983) Deterministic parsing of syntactic non-fluencies. In *Proceedings of the 21st Annual Meeting of the ACL*, 123–8.

Hindle, D. (1989) Acquiring disambiguation rules from text. In *Proceedings of the 27th Annual Meeting of the ACL*, 118–25.

Hindle, D. and Rooth, M. (1993) Structual ambiguity and lexical relations, *Computational Linguistics* 19(1) : 103–20.

Hofland, K. (1991) Concordance programs for personal computers. In Johansson and Stenström (1991), 283–306.

Hogenraad, R., Bestgen, Y., and Nysten, J.-L. (1995). Terrorist rhetoric: texture and architecture. In E. Nissan and K. M. Schmidt (eds), *From information to knowledge: conceptual and content analysis by computer.* Oxford: Intellect, 48–59.

Holsti, O. (1969) *Content analysis for the social sciences.* Reading, MA: Addison Wesley.

Hughes, L. and Lee, S. (1994) (eds) *CTI Centre for Textual Studies resources guide (1994)*, Oxford: CTI Centre for Textual Studies.

Ide, N. (1996) Corpus Encoding Standard. MULTEXT/EAGLES – Document CES 1. Version 1.4. Nancy Ide, Coordinator

Ide, N. and Véronis, J. (1995) (eds) *Text Encoding Initiative.* Dordrecht: Kluwer.

Jackson, H. (1990) *Grammar and meaning: A semantic approach to English grammar.* London: Longman.

Janssen, S. (1990) Automatic sense disambiguation with LDOCE: Enriching syntactically analyzed corpora with semantic data. In J. Aarts and W. Meijs (eds) *Theory and practice in corpus linguistics.* Amsterdam: Rodopi, 105–35.

Jelinek, F. (1976) 'Continuous speech recognition by statistical methods.' In *Proceedings of the IEEEE* 64(4), 532–56.

Jelinek, F. (1985) Markov source modeling of Text generation. In J. K. Skwirzinski (ed.) *Impact of processing techniques on communication.* Dordrecht: Nijhoff.

Jelinek, F. (1990) Self-organized language modeling for speech recognition. In A. Waibel and K-F Lee (eds) *Readings in speech recognition.* San Mateo, CA: Morgan Kaufmann, 450–506.

Jelinek, F. and Lafferty, J. D. (1991) Computation of the probability of initial substring generation by stochastic context-free grammars, *Computational Linguistics* 17: 3, 315–23.

Johansson, S. (1986) *The Tagged LOB Corpus: users' manual.* Bergen: Norwegian Computing Centre for the Humanities.

Johansson, S. (1995) The approach of the Text Encoding Initiative to the encoding of spoken discourse. In Leech *et al.* (1995) 82–98.

Johansson, S. and Stenström, A.-B. (1991) (eds) *English computer corpora: Selected papers and research guide.* Berlin: Mouton de Gruyter.

Johansson, S. Ebeling, J., and Hofland, K. (1993) Coding and aligning the English-Norwegian parallel corpus. In K. Aijmer, B. Altenberg, and M. Johansson (eds) *Languages in contrast: a symposium on text-based cross-linguistic studies.* Lund Studies in English 88. Lund: Lund University Press, 87–112.

Johansson, S. Leech, G., and Goodluck, H. (1978) *Manual of information to accompany the Lancaster-Oslo/Bergen Corpus of British English, for use with digital computers.* Department of English, University of Oslo.

Kabanoff, B., Waldersee, R., and Cohen, M. (1995). Espoused values and organizational-change themes. *Academy of Management Journal* 38: 4, 1075–104.

Källgren, G. (1996) 'Linguistic indeterminacy as a source of errors in tagging.' In *Proceedings of the 16th international conference on computational linguistics (COLING)*, Copenhagen.

Karlsson, F. (1994) Robust parsing of unconstrained text. In Oostdijk and de Haan (1994), 121–42.

Karlsson, F., Voutilainen, A., Heikkilä, J., and Anttila, A. (1995) (eds) *Constraint Grammar, a language-independent system for parsing unrestricted text.* Berlin and New York: Mouton de Gruyter.

Kazanskiene, V. P. and Kazanskij, N. N. (1986) *Dictionnaire idéologique de la langue grecque. Période créto-mycénienne.* Leningrad: Nauka.

Kelly, E. F. and Stone, P. J. (1975) *Computer recognition of English word senses.* Amsterdam: North-Holland.

Kiesler, D. J. (1973) *The process of psychotherapy: empirical foundations and systems of analysis.* Chicago, Ill.: Aldine.

Kingdon, R. (1958) *The groundwork of English intonation.* London: Longman.

Kirk, J. M. (1994) Taking a byte at Corpus Linguistics. In L. Flowerdew and A. K. K. Tong (eds), *Entering Text.* Language Centre, The Hong Kong University of Science and Technology. 18–49.

Knowles, G. (1991) Prosodic labelling: The problem of tone group boundaries. In Johansson and Stenström (1991), 149–63.

Knowles, G. (1993) The machine-readable Spoken English Corpus. In Aarts *et al.* (1993), 107–19.

Knowles, G. (1995) Converting a corpus into a relational database. In Leech *et al.* (1995), 208–19.

Knowles, G., Wichmann, A. and Alderson, P. (1996b) *Working with speech: perspectives on research into the Lancaster/IBM Spoken English Corpus.* London: Longman.

Knowles, G., Williams B. and Taylor, L. (1996a) *A corpus of formal British English speech: the Lancaster/IBM Spoken English Corpus.* London: Longman.

Kupiec, J. M. (1989) Probabilistic models of short and long distance word dependencies in running texts. In *Proceedings of the (1989) DARPA Speech and Natural Language Workshop*, Philadelphia: Morgan Kaufman, 290–5.

Lancashire, I. (1991) (ed.) *The Humanities Computing Yearbook (1989-90).* Oxford: Oxford University Press.

Langendoen, D. T. and Fahmy, E. (1991). *Feature-structure markup for presentation at Oxford and Brown workshops*, Department of Linguistics, University of Arizona.

Lasswell, H. D., Leites, N., and associates (1949/1965) *Language of Politics: Studies in quantitative semantics.* Cambridge, MA: MIT Press.

LDOCE (1978). *Longman dictionary of contemporary English.* (1st edition) London: Longman.

Leech, G. N. (1991) The state of the art in corpus linguistics. In Aijmer K. and Altenberg B. (eds) *English corpus linguistics: studies in honour of Jan Svartvik*, London: Longman.

Leech, G. N. (1994) 100 million words of English: the British National Corpus, *English Today* 9(1), 9–15.

Leech, G. N. and Fligelstone, S. (1992) Computers and corpus analysis. In C. S. Butler (ed.) *Computers and written texts*, Oxford: Blackwell, 115–40.

Leech, G. N. and Garside, R. (1991) 'Running a grammar factory: the production of syntactically analysed corpora or "treebanks".' In Johansson and Stenström (1996), 15–32.

Leech, G. N. and Short, M. H. (1981) *Style in fiction: a linguistic introduction to English fictional prose.* London: Longman.

Leech, G. N. and Wilson, A. (1994) *EAGLES Morphosyntactic annotation.* EAGLES Report EAGCSG/IR-T3. 1. Pisa: Istituto di Linguistica Computazionale.

Leech, G. N., Barnett, R., and Kahrel, P. (1995) *Guidelines for the standardization of Syntactic Annotation of Corpora.* EAGLES Document EAG-TCWG-SASG/1.8.

Leech, G. N., Deuchar, M., and Hoogenraad, R. (1982) *English grammar for today.* London: Macmillan.

Leech, G. N., Francis, B., and Xu, X. (1994) The use of computer corpora in the textual demonstrability of gradience in linguistic categories. In C. Fuchs and B. Victorri (eds) *Continuity in linguistic semantics*, Amsterdam: Benjamins, 57–76.

Leech, G. N., Garside, R., and Bryant, M. (1994) The large-scale grammatical tagging of text: Experience with the British National Corpus. In Oostdijk and de Haan (1994), 47–63.

Leech, G. N., Myers, G., and Thomas, J. (1995) *Spoken English on computer: Transcription, mark-up and application.* London: Longman.

Lesk, M. E. (1986) 'Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone.' *Proceedings of the (1986) SIGDOC Conference*, New York: ACM.

Li, C. N. (1986) 'Direct speech and indirect speech: a functional study.' In Coulmas (1986).

Louw, J. P., Nida, E. A. (1989) *Greek-English lexicon of the New Testament, based on semantic domains.* 2 vols. New York: United Bible Societies.

McArthur, T. (1981) *Longman lexicon of contemporary English.* London: Longman.

McArthur, T. (1986) *Worlds of reference.* Cambridge: Cambridge University Press.

McEnery, A. M. (1995) Computational pragmatics, Ph.D Thesis, Lancaster University.

McEnery, A. M. (1997) *Computing and pragmatics.* New Jersey: Ablex.

McEnery, A. M. and Daille, B. (1993) *Database design for corpus storage: the ET10–63 Data Model.* UCREL Technical paper 1. Lancaster University, UK.

McEnery, A. M. and Oakes. M. P. (1995) Sentence and word alignment in the CRATER project: Methods and assessment, in *Proceedings of the European Chapter of the Association of Computational Linguistics EACL-SIGDAT Workshop*, Dublin.

McEnery, A. M. and Oakes, M. P. (1996) Sentence and word alignment in the CRATER project. In Thomas and Short (1996), 211–31.

McEnery, A. M. and Wilson, A. (1994) 'The role of corpora in computer assisted language learning.' *Computer Aided Language Learning* 6: 3, 233–48.

McEnery, A. M. and Wilson, A. (1996) *Corpus linguistics*, Edinburgh: Edinburgh University Press.

McEnery, A. M. Baker, P. and Wilson, A. (1995) 'A statistical analysis of corpus based computer vs traditional human teaching methods of part of speech analysis.' *Computer Assisted Language Learning* 8 (2/3): 259–74.

McEnery, A. M. Oakes, M. P., and Garside, R. (1994) The use of approximate string matching techniques in the alignment of sentences in parallel corpora. In *Proceedings of Machine Translation: 10 Years On*, Cranfield University.

McHale, B. (1978) Free indirect speech: a survey of recent accounts, *Poetics and The Theory of Literature*, 3: 249–88.

MacWhinney, B. (1991) *The CHILDES project: tools for analyzing talk.* Hillsdale, NJ: Lawrence Erlbaum.

MacWhinney, B. and Snow, C. (1990) 'The Child Language Data Exchange System', *ICAME Journal* 14, 3–25

de Marcken, C. (1990) 'Parsing the LOB Corpus.' In *Proceedings of the 28th International meeting of the Association for Computational Linguistics*, 243–251.

Marcus, M. (1995) New trends in natural language processing: Statistical natural language processing, *Proceedings of the National Academy of Science USA*, 92, 10052–59.

Marcus, M. and Santorini, B. (1992). *Building very large natural language corpora: the Penn Treebank*, Department of Computer and Information Science, University of Pennsylvania.

Marcus, M., Santorini, B, Marcinkiewicz, M. (1993) Building a large annotated corpus of English: the Penn Treebank, *Computational Linguistics* 19(2), 313–30.

Marshall, I. (1983) Choice of grammatical word-class without global syntactic analysis: tagging words in the LOB Corpus, *Computers and the Humanities* 17, 139–50.

Martindale, C. (1974) The semantic significance of spatial movement in narrative verse: Patterns of regressive imagery in the Divine Comedy. In L. Mitchell (ed.) *Computers in the humanities*. Edinburgh: Edinburgh University Press, 57–64.

Merialdo, B. (1994) 'Tagging English text with a probabilistic model.' *Computational Linguistics* 20(2), 155–71.

Miller, G., Beckwith, R., Fellbaum, C., Gross, D., Miller, K., Tengi, R. (1990/93) *Five papers on WordNet.* CSL Report 43, Cognitive Science Laboratory, Princeton University.

Mitkov, R. (1994): A new approach for tracking center. In *Proceedings of the International Conference 'New Methods in Language Processing'*, Manchester: UMIST.

Mitkov, R. (1995) Anaphora resolution in natural language processing and machine translation. Saarbrücken: IAI Working Paper.

Mitkov, R. (forthc.) Two engines are better than one: generating more power and confidence in the search for the antecedent. In R. Mitkov and N. Nicolov (eds) *Recent advances in natural language processing*. Amsterdam: Benjamins.

Nagao, M. (1984) 'A framework of a mechanical translation between Japanese and English by Analogy Principle.' In A. Elithorn and R. Bannerji (eds), *Artificial and Human Intelligence*, Amsterdam: North-Holland, 173–80.

Najock, D. (forthc.) *Ein statistischer Schlüssel zum Vokabular in Vergils Eklogen*, Hildesheim: Georg Olms Verlag.

Nederhof, M.-J. and Koster, K. (1993) 'A customized grammar workbench.' In Aarts *et al.* (1993), 163–179.

Oakes, M. P. and Taylor, M. J. (1992) 'Clustering of thesaurus terms using ART, FCMs and approximate string-matching techniques.' In R. Beale and J. Findlay (eds) *Neural networks and pattern recognition in human-computer interaction*, Chichester: Ellis Horwood, 246–63.

Ochs, E. (1979) Transcription as theory. In E. Ochs and B. B. Schiffelin (1979) *Developmental pragmatics*. New York: Academic Press, 43–72.

Oostdijk, N. and de Haan, P. (1994) (eds) *Corpus-based research into language*. Amsterdam: Rodopi.

Oxford University Computing Services (1995) *Users Reference Guide for the British National Corpus Version 1.0.*

Pennington, M. and Stevens, V. (1992) (eds) *Computers in applied linguistics.* Clevedon: Multilingual Matters.

Peppé, S. (1995) The Survey of English Usage and the London-Lund Corpus: Computerizing manual prosodic transcription. In Leech *et al.* (1995), 187–202.

Pickering, B., Williams, B., and Knowles, G. (1996) 'Analysis of transcriber differences in the SEC.' In Knowles (1996b), 61–86.

Pierrehumbert, J. (1980) The phonology and phonetics of English intonation. Ph.D. Dissertation, MIT. Distributed by the Indiana University Linguistics Club.

Poritz, A. B. (1988) 'Hidden Markov models: a guided tour.' In *International Conference on Acoustics, Speech and Signal Processing*, vol i, 7–13. New York: IEEE.

Pustejovsky, D. (1995) *The generative lexicon.* Cambridge, MA: MIT Press.

Quinn A. (1993) 'An object-oriented design for a Corpus Utility Program.' In Aarts *et al.* (1993), 215–25

Quirk, R. (1960) Towards a description of English usage, *Transactions of the Philological Society*, 40–61.

Quirk, R. and Greenbaum, S. (1973) *A university grammar of English*, London: Longman.

Quirk, R., Greenbaum, S., Leech, G., and Svartvik, J. (1972) *A grammar of contemporary English.* London: Longman.

Quirk, R., Greenbaum, S., Leech, G., and Svartvik, J. (1985) *A comprehensive grammar of the English language.* London: Longman.

Rayson, P. and Wilson, A. (1996) [Reference deleted for reasons of commercial confidentiality]

Rayson, P., Leech, G. N., and Hodges, M. (forthc.). Social differentiation in the use of English vocabulary: some analyses of the conversational component of the British National Corpus. *International Journal of Corpus Linguistics.*

Roach P. and Arnfield S. (1995) Linking prosodic transcriptions to the time dimension. In Leech *et al.* (eds), 149–60.

Roberts, J. A. (1978) 'Towards an Old English Thesaurus.' *Poetica*, 9, 56–72.

Roberts, J. A. (1994) Old English thesaurus: progress and plans. In M. Kytö, M. Rissanen, and S. Wright (eds) *Corpora across the centuries.* Amsterdam: Rodopi.

Robertson, A. and Willett, P. (1993) Evaluation of techniques for the conflation of modern and seventeenth century English Spelling. In A. McEnery and C. Paice (eds) *Proceedings of the 14th Information Retrieval Colloquium.* London: Springer Verlag, 155–68.

de Rocha, M. (1996) A corpus-based study of anaphora in English and Portuguese. In Botley and McEnery (forthc.).

Roget, P. M. (1852) *Thesaurus of English words and phrases.* London: Longman, Brown, Green.

Russell, R. L. (1988) 'A new classification scheme for studies of verbal behaviour in psychotherapy', *Psychotherapy* 25.

Sampson, G. (1987a) Probabilistic models of analysis. In Garside *et al.* (1987), 16–29.

Sampson, G. (1987b) The grammatical database and parsing. In Garside *et al.* (1987), 82–96.

Sampson, G. (1987c) Alternative grammatical coding systems. In Garside *et al.* (1987), 165–83.

Sampson, G. (1992) Analysed corpora of English: a consumer guide. In Pennington and Stevens (1992), 181–200.

Sampson, G. (1994) SUSANNE: A Domesday Book of English grammar. In N. Oostdijk and de Haan (1994), 169–88.

Sampson, G. (1995) *English for the computer: The SUSANNE Corpus and analytic scheme.* Oxford: Clarendon Press.

Sánchez León, F. (1995) *Spanish tagset for the CRATER project*, CRATER Internal Document, Final version.

Sánchez León, F. and Nieto Serrano, A. (1995) *Public Domain POS Tagger for Spanish*, CRATER Final Deliverable Report No. 22.

Sanfilippo, A., *et al.* (1996) EAGLES, lexicon syntax document.

Santorini, B. (1990) *Part of speech tagging guidelines for the Penn Treebank Project.* Philadelphia: Technical Report MS-CIS-90–47, Department of Computer and Information Science, University of Pennsylvania.

Schlessing, A. (1881) *Deutscher Wortschatz oder der passende Ausdruck.* Stuttgart: Neff.

Schmidt, K. M. (1986) Concept versus meaning: the contribution of computer-assisted content analysis and conceptual analysis to this disputed area. In *En hommage à Charles Muller: méthodes quantitatives et informatiques dans l'étude des textes.* Geneva: Slatkine, 779–93.

Schmidt, K. M. (1988) Der Beitrag der begriffsorientierten Lexikographie zur systematischen Erfassung von Sprachwandel und das Begriffswörterbuch zur mhd. Epik. In Bachofer W. (ed.) *Mittelhochdeutsches Wörterbuch in der Diskussion.* Tübingen: Max Niemeyer, 35–49.

Schmidt, K. M. (1991) Ein Datenbanksystem für das Begriffswörterbuch mittelhochdeutscher Epik und Fortschritte bei der automatischen Disambiguierung. In Gärtner K., Sappler P., Trauth M. (eds) *Maschinelle Verarbeitung altdeutscher Texte IV.* Tübingen: Max Niemeyer, 192–204.

Schmidt, K. M. (1993) *Begriffsglossar und Index zu Ulrichs von Zatzikhoven Lanzelet.* Tübingen: Max Niemeyer.

Schutz, J. (1994) Terminological Knowledge in Multilingual Language Processing. *Studies in Machine Translation and Natural Language Processing*, Volume 5. Office for Official Publications of the European Community, Luxembourg, 1994.

Scott, M. R. (1996) *WordSmith Tools.* Oxford: Oxford University Press.

Sedelow, S. and Sedelow, W. (1969). Categories and procedures for content analysis in the humanities. In G. Gerbner, O. R. Holsti, K. Krippendorff, W. J. Paisley, and P. J. Stone (eds) *The analysis of communication content.* New York: John Wiley, 487–99.

Semino, E., Short, M., and Culpeper, J. (forthc.), Using a computer corpus to test and refine a model of speech and thought presentation, *Poetics.*

Short, M., Semino, E., and Culpeper, J. (1996) Using a corpus for stylistics research: speech and thought presentation. In Thomas and Short (1996), 110–31.

Sidner, C. (1986) Focusing in the comprehension of definite anaphora. In B. Grosz, K. Jones, and B. Webber (eds) *Readings in natural language processing*, Los Altos, CA: Morgan Kaufmann Publishers.

Simard, M., Foster, G. and Isabelle, P. (1992). Using cognates to align sentences in bilingual corpora.' *Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI9)*, Montreal, Canada. 67–81.

Simons, G. F. (1991) *Feature System Declarations and the Interpretation of Feature Structures*, January (1991).

Sinclair, J. (1992) The automatic analysis of corpora. In Svartvik (1992), 379–97.

Sinclair, J. and Coulthard, R. (1975) *Towards an analysis of discourse: the English used by teachers and pupils*. London: Oxford University Press.

Smadja, F. (1993) Retrieving collocations from text: Xtract. *Computational Linguistics* 19 (1), 143–77.

Smadja, F. (1989) *A short handbook to the Polytechnic of Wales Corpus*. Bergen: Norwegian Computing Centre for the Humanities.

Smadja, F. (1993) Towards a standard format for parsed corpora. In Aarts *et al.* (1993), 197–212.

Smadja, F. and Atwell, E. (1994) Using parsed corpora: A review of current practice. In Oostdijk and de Haan (1994), 143–58.

Sperberg-McQueen, C. M. (1991) Texts in the electronic age: textual study and text encoding, with examples from medieval text. *Literary and Linguistic Computing* 6(1) : 34–6.

Sperberg-McQueen, C. M. and Burnard, L. (1994) (eds) *Guidelines for electronic text encoding and interchange*, Document Number: TEI-P3. Chicago and Oxford: ACH, ACL, ALLC.

Steel, D. and Alderson, J. C. (1994) 'Metalinguistic Knowledge, Language Aptitude and Language Proficiency', *CRILE Report Number 5*, Department of Linguistics, Lancaster University.

Stenström, A.-B. (1984) Discourse tags. In J. Aarts and W. Meijs (eds) *Corpus linguistics: Recent developments in the use of computer corpora in English language research*, Amsterdam: Rodopi, 65–81.

Stiles, W. B. (1992) *Describing talk: a taxonomy of verbal response modes*. Beverly Hills: Sage.

Stolcke, A. (1995) An efficient probabilistic context free parsing algorithm that computes prefix probabilities, *Computational Linguistics* 21: 2, 165–201.

Stone, P. J. *et al.* (1966) *The general inquirer: a computer approach to content analysis*. Cambridge, MA: MIT Press.

Svartvik, J. (1990) (ed.) *The London-Lund Corpus of spoken English: description and research*. Lund Studies in English 82. Lund: Lund University Press.

Svartvik, J. (1992) (ed.) *Directions in Corpus Linguistics. Proceedings of Nobel Symposium 82, Stockholm*. Berlin and New York: Mouton de Gruyter.

Svartvik, J. and Eeg-Olofsson, M. (1982) Tagging the London-Lund Corpus of Spoken English. In S. Johansson (ed.) *Computer corpora in English language research*, Bergen: Norwegian Computing Centre for the Humanities, 85–109.

Svartvik, J. and Quirk, R. (1980) (eds) *A Corpus of English Conversation.* Lund Studies in English 56. Lund: Gleerup/Liber.

Tapanainen, P. and Voutilainen, A. (1994) Tagging accurately – Don't guess if you know. In *Proceedings of the Fourth Conference on Applied Natural Language Processing,* Stuttgart.

Taylor, L. and Knowles, G. (1988) Progress report on the Lancaster Spoken English Corpus. *ICAME Journal,* 12, 62–3.

Teufel, S. (1995) 'Abbilding zwischen Annotationsschema zur Unterstützung von morphosyntaktischen Standards.' In W. Hoetker and P. Ludewig (eds), *Lexikonimport, Lexikonexport.* Tübingen: Niemeyer.

Text Encoding Initiative (1991) *TEI AI1W2. List of common morphological features for inclusion in TEI starter set of grammatical-annotation tags.*

Thomas, J. (1996) *Meaning in interaction: an introduction to pragmatics.* London: Longman.

Thomas, J. and Short, M. (1996) (eds) *Using corpora for language research.* London: Longman.

Thomas, J. and Wilson, A. (1996) Methodologies for studying a corpus of doctor–patient interaction. In Thomas and Short (1996), 92–109.

Thompson, H. and McKelvie (1996) A Software Architecture for SGML Annotation. *Proceedings of SGML Europe 96.* SoftQuad.

Trager, G. L. and Smith, H. L. (1951) *Outline of English structure.* Norman, OK: Battenburg Press.

Tuthill, W. (1981) *HUM: a concordance and text analysis package.* Unpublished on-line Help package. University of California at Berkeley.

Ungerer, F. and Schmid, H.-J. (1996) *An introduction to cognitive linguistics.* London: Longman.

van Halteren, H. (forthc.) *Syntactic wordclass tagging.* Berlin: Mouton de Gruyter.

van Halteren, H. and van den Heuvel, Th. (1990) *Linguistic exploitation of syntactic databases: The use of the Nijmegen Linguistic DataBase Program.* Amsterdam: Rodopi.

van Halteren, H. and Oostdijk, N. (1993) Towards a syntactic database: The TOSCA analysis system. In Aarts *et al.* (1993), 145–62.

Véronis, J. (1996) Guidelines for Linguistic Software Development GLOSIX Version 0. MULTEXT/EAGLES – Document LSD 2. Jean Véronis, Coordinator.

Véronis, J. and Ide, N. (1996) Considerations for the Reusability of Linguistic Software. MULTEXT/EAGLES – Document LSD 1. Jean Véronis, Coordinator.

Viney, L. L. (1983) 'The assessment of psychological states through content analysis of verbal communication', *Psychological Bulletin* 94.

Voutilainen, A. (1994) *Three studies of grammar-based surface parsing of unrestricted English text.* Publication No. 24, Department of General Linguistics, University of Helsinki.

Voutilainen, A. (1995) Morphological disambiguation. In Karlsson *et al.* (1995) 165–283.

Voutilainen, A. and Jarvinen, T. (1995) Specifying a shallow grammatical representation for parsing purposes. In *Proceedings of the Seventh Conference of the European Chapter of the Association for Computational Linguistics.*

Voutilainen, A. Heikkilä, J., and Anttila, A. (1992) *Constraint grammar of English: A performance-oriented introduction.* Publications No. 21, Department of General Linguistics, University of Helsinki.

Weber, R. P. (1982) The long-term problem-solving dynamics of social systems. *European Journal of Political Research* 10, 387–405.

Weber, R. P. (1989). *Basic Content Analysis.* Sage University Paper series on Quantitative Applications in the Social Sciences, no. 07–049. Newbury Park, CA: Sage.

Weischedel, R., Meteer, M., Schwartz, R., Ramshaw, L., and Palmucci, J. (1993) 'Coping with ambiguity and unknown words through probabilistic models.' *Computational Linguistics* 19/2, 359–82.

Wherle, H. and Eggers, H. (1961) *Deutscher Wortschatz: ein Wegweiser zum treffenden Ausdruck.* Stuttgart: Ernst Klett.

Wichmann, A., Fligelstone, S., Knowles, G., and McEnery, A. M. (1997) *Teaching and language corpora.* London: Longman.

Wiebe, J. (1994), Recognising subjective sentences: a computational investigation of narrative text. Unpublished Ph.D. thesis, SUNY Buffalo.

Wiebe, J. and Rappaport, W. J. (1991) Representing *de re* and *de dicto* belief reports in discourse and narrative, *Proceedings of the IEEE* 74, 1405–13.

Williams, B. (1996a) The status of corpora as linguistic data. In Knowles *et al.* (1996b), 1–19.

Williams, B. (1996b) The formulation of an intonation transcription system for British English. In Knowles *et al.* (1996b), 38–57.

Wilson, A. (1991) *No, not,* and *never* negation in a corpus of spoken interview transcripts. *Lancaster papers in Linguistics* 73.

Wilson, A. (1993) Towards an integration of content analysis and discourse analysis: the automatic linkage of key relations in text. *UCREL Technical Paper* 3, Lancaster University.

Wilson, A. (1996) Conceptual glossary and index to the Latin vulgate translation of the Gospel of John. Ph.D. dissertation, Lancaster University.

Wilson, A. and Leech, G. N. (1993) Automatic content analysis and the stylistic analysis of prose literature. *Revue Informatique et Statistique dans les Sciences Humaines* 29: 219–34.

Wilson, A. and Rayson, P. (1993) [Reference deleted for reasons of commercial confidentiality]

Wu, Z. and Tseng, G. (1993) 'Chinese text segmentation for text retrieval: achievements and problems.' *Journal of the American Society for Information Science* 44 (9): 532–42.

Yarowsky, D. (1992) Word-Sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora. *Proceedings of COLING 92.*

Zadeh, L. (1982) A note on prototype theory and fuzzy sets. *Cognition* 12, 291–9.

Züll, C., Mohler, P. Ph., and Weber, R. P. (1989) *Computer assisted text classification for the social sciences: the General Inquirer III.* Mannheim: ZUMA.

Index