Unit 4 Corpus annotation

4.1 Introduction

Corpus annotation is closely related to corpus markup. One important reason for using corpora in linguistic research is to extract linguistic information present in those corpora. But it is often the case that in order to extract such information from a corpus, a linguistic analysis must first be encoded in the corpus. The process of 'adding such interpretative, linguistic information to an electronic corpus of spoken and/or written language data' is referred to as corpus annotation (Leech 1997a: 2). Corpus annotation adds value to a corpus in that it considerably extends the range of research questions that a corpus can readily address. While corpus annotation defined in a broad sense may refer to the encoding of both textual/contextual information and interpretative linguistic analysis, as shown by the conflation of the two often found in the literature, the term is used in a narrow sense here, referring solely to the encoding of linguistic analyses such as part-of-speech (POS) tagging and syntactic parsing in a corpus text.

Corpus annotation, as used in a narrow sense, is fundamentally distinct from corpus markup as discussed in unit 3. Corpus markup provides relatively objectively verifiable information regarding the components of a corpus and the textual structure of each text. In contrast, corpus annotation is concerned with interpretative linguistic information. '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' (Leech 1997a: 2). For example, the part-of-speech of a word may be ambiguous and hence is more readily defined as corpus annotation than corpus markup. On the other hand, the sex of a speaker or writer is normally objectively verifiable and as such is a matter of markup, not annotation.

This unit will first discuss the advantages and disadvantages of corpus annotation. Following this is a discussion of how corpus annotation is achieved. We will then introduce the most commonly used types of corpus annotation. Finally we will briefly review stand-alone corpus annotation, as proposed by the Corpus Encoding Standard (CES, see unit 3.3).

4.2 Corpus annotation = added value

Like corpus markup, annotation adds value to a corpus. Leech (1997a: 2) maintains that corpus annotation is '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.' Both Leech (*ibid*: 4-5) and McEnery (2003: 454-455) suggest that there are at least four advantages for corpus annotation.

Firstly, it is much easier to extract information from annotated corpora in a number of ways. Leech (*ibid*) observes, for example, that without part-of-speech tagging (see unit 4.4.1), it is difficult to extract *left* as an adjective from a raw corpus as its various meanings and uses cannot be identified from its orthographic form or context alone. For example, the orthographic form *left* with a meaning opposite to *right* can be an adjective, an adverb or a noun. It can also be the past or past participle form of *leave*. With appropriate part-of-speech annotations these different uses of *left* can be readily distinguished apart. Corpus annotation also enables human analysts and machines to exploit and retrieve analyses of which they are not themselves capable

(McEnery 2003: 454). For example, even if you do not know Chinese, given a suitably annotated Chinese corpus, you are able to find out a great deal about Chinese using that corpus (see case study 6 in Section C). Speed of data extraction is another advantage of annotated corpora. Even if one is capable of undertaking the required linguistic analyses, one is quite unlikely to be able to explore a raw corpus as swiftly and reliably as one can explore an annotated corpus if one has to start by annotating the corpus oneself.

Secondly, an annotated corpus is a reusable resource, as annotation records linguistic analyses within the corpus that are then available for reuse. Considering that corpus annotation tends to be costly and time consuming, reusability is a powerful argument in favour of corpus annotation (cf. Leech 1997a: 5).

Thirdly, an advantage of corpus annotation, related to reusability, is multi-functionality. A corpus may have originally been annotated with one specific purpose in mind. However, corpus analyses may be reused for a variety of applications and even for purposes not originally envisaged.

Finally, corpus annotation records a linguistic analysis explicitly. As such, the corpus annotation stands as a clear and objective record of analysis that is open to scrutiny and criticism (cf. McEnery 2003), a laudable goal.

In addition to these advantages we can also note that corpus annotation, like a corpus *per se*, provides a standard reference resource. While a corpus may constitute a standard reference for the language variety which it is supposed to represent, corpus annotation provides a stable base of linguistic analyses, objectively recorded, so that successive studies can be compared and contrasted on a common basis.

Having outlined the advantages of corpus annotation, it is necessary to address some of the criticisms of corpus annotation. Four main criticisms of corpus annotation have been presented over the past decade.

The first criticism is that corpus annotation produces cluttered corpora. Hunston (2002: 94) argues that '[h]owever much annotation is added to a text, it is important for the researcher to be able to see the plain text, uncluttered by annotational labels.' While we agree that the plain text is important in a corpus analysis, especially in observing the patterning of words, corpus annotation does not necessarily obscure such a patterning, because most corpus exploration tools (e.g. WordSmith, MonoConc, SARA and Xaira, see Section C) do indeed allow users to suppress annotation in search results so as to allow users to view the plain text. As such this criticism is more directed at corpus browsing/retrieval tools rather than at corpus annotation *per se*.

A second criticism is that annotation imposes a linguistic analysis upon a corpus user. While it is true that corpus annotation is fundamentally interpretative in nature, there is no compulsion that corpus users accept that analysis. They can impose their own interpretations if they will or simply ignore the annotation. The plurality of interpretations of a text is something that must be accepted from the outset when undertaking corpus annotation (cf. McEnery 2003: 456). Yet just leaving a corpus unannotated does not mean that there is no process of interpretation occurring when the corpus is analyzed. Rather, the lack of annotation simply disguises the fact that such multiple-interpretations still occur when researchers use a raw corpus. The analysis still happens, it is simply hidden from clear view. Corpus annotation should be recognized as an advantage rather than a weakness in this respect as it provides an objective record of an explicit analysis open for scrutiny – failing to annotate is not simply a failure to analyze. Failing to annotate does, however, ensure that the analysis is difficult, or indeed impossible, to recover.

A further criticism is that annotation may 'overvalue' a corpus, making it less readily accessible, updateable and expandable (cf. Hunston 2002: 92-93). Annotation does not necessarily makes a corpus less accessible. For example, many parsed (e.g. the Lancaster Parsed Corpus and the Susanne corpus, see unit 7.4) and prosodically annotated corpora (e.g. the London-Lund Corpus and the Lancaster/IBM Spoken English Corpus, see unit 7.5) are publicly available. Corpus builders are usually happy to make their corpora available as widely as possible in spite of (or sometimes because of) the huge effort that they have put into annotation. Funders are also often prepared to finance corpus construction because a valuable annotated resource will be made widely available. Public funding bodies are particularly unlikely to fund corpus building projects which do not result in a readily accessible resource. A more common reason for not making an annotated corpus (or indeed a raw corpus) publicly available is that the copyright issues related to the corpus data prohibit it (see unit 9). Copyright, not annotation, is the greater force in favour of restriction. The arguments relating to updating and expansion are also questionable. Unlike a monitor corpus, which is constantly updated to track rapid language change (see unit 7.9 for further discussion), most corpora are sample corpora. A sample corpus is designed to represent a particular language variety at a particular time. For example, the LOB and Brown corpora are supposed to represent written British and American English in the early 1960s. There are indeed 'updates' for the two corpora – FLOB and Frown (see unit 7.4). The two updated corpora respectively represent written British and American English in the early 1990s and can be used to track slower paced language change (see unit 15.5). The need for constant expansion is only related to the dynamic monitor corpus model. It does not necessarily apply as an argument to sample corpora. Given that most corpora are sample corpora, the expandability argument is hardly important, as with a sample corpus size is typically determined when the corpus is designed. Once the corpus is created, there is generally no need for expansion.

The final criticism is related to the accuracy and consistency of corpus annotation. There are three basic methods of annotating a corpus – automatic, computer-assisted and manual (see unit 4.3). On the one hand, as Hunston (2002: 91) argues, 'an automatic annotation program is unlikely to produce results that are 100% in accordance with what a human researcher would produce; in other words, there are likely to be errors.' Such errors also occur when humans alone analyze the texts – even the best linguist at times makes mistakes. Introducing a human factor into annotation may have another implication; Sinclair (1992) argues, the introduction of a human element in corpus annotation, as in manual or computer-assisted annotation, results in a decline in the consistency of annotation. Taking the two points together, one might wonder why any linguist has ever carried out an analysis, as it would have been inaccurate and inconsistent! One must conclude that they have done so, and that annotators continue to do so because while inconsistency and inaccuracy in analyses are indeed observable phenomena, their impact upon an expert human analysis has been exaggerated. Also, the computer is not a sure-fire means of avoiding inaccuracy or inconsistency: the two points may also apply to machine analyses. Automatic annotation is not error-free, and it may be inconsistent. If resources are altered for an annotation program – the lexicon changed, rules rewritten – then over time the output of the program will exhibit inconsistency on a scale that may well exceed that displayed by human analysts. So what should we use for corpus annotation, human analysts or the computer? Given that the value of corpus annotation is well recognized, the human analyst and the machine should complement each other, providing a balanced approach to accuracy and consistency that seeks to reduce inaccuracy and

inconsistency to levels tolerable to the research question that the corpus is intended to investigate.

It is clear from the above discussion that all of the four criticisms of corpus annotation can be dismissed, with caveats, quite safely. Annotation only means undertaking and making explicit a linguistic analysis. As such, it is something that linguists have been doing for centuries.

4.3 How is corpus annotation achieved?

As noted earlier, corpus annotation can be achieved fully automatically, by a semi-automatic interaction between human being and the machine, or entirely manually by human analysts. To cover the three in turn, in automatic annotation, the computer works alone as an annotator by following the rules and algorithms predefined by a programmer, though rules can also be acquired by the machine via machine learning (ML) using a predefined ML algorithm. Developing an automatic annotation tool may cost time and money, but once it is completed, large quantities of data can be annotated rapidly and (assuming there are no resource changes) consistently. On occasion one may find that this work has already been undertaken elsewhere and that a program that can undertake the desired annotation is already freely available.

Some types of annotation, e.g. lemmatization and POS tagging (see units 4.4.1 and 4.4.2) for English, French and Spanish, and segmentation and POS tagging for Chinese, can be undertaken by the machine so reliably (with a typical error rate of 3%) that we may consider a wholly automated approach to their annotation. When the output from an automated process is not reliable, as is the case with most parsers (see unit 4.4.5), or the output is reliable, but not accurate enough for a particular purpose (e.g. the training corpus used to improve an annotation tool), human correction (i.e. post-editing) is usually required. Post-editing is generally faster than undertaking annotation entirely by hand. Some annotation tools provide a human-machine interface that allows a human analyst to resolve ambiguous cases where the machine is not certain. The semi-automatic annotation process may produce more reliable results than fully automated annotation, but it is also slower and more costly. Pure manual annotation occurs where no annotation tool is available to a user, or where the accuracy of available systems is not high enough to make the time invested in manual correction less than pure manual annotation. As manual annotation is expensive and time-consuming, it is typically only feasible for small corpora. With the few exceptions mentioned above, most types of annotation presently available in large corpora were introduced either semi-automatically or manually.

4.4 Types of corpus annotation

Corpus annotation can be undertaken at different levels and may take various forms. For example, at the phonological level corpora can be annotated for syllable boundaries (phonetic/phonemic annotation) or prosodic features (prosodic annotation); at the morphological level corpora can be annotated in terms of prefixes, suffixes and stems (morphological annotation); at the lexical level corpora can be annotated for parts-of-speech (POS tagging), lemmas (lemmatization), semantic fields (semantic annotation); at the syntactic level corpora can be annotated with syntactic analysis (parsing, treebanking or bracketing); at the discoursal level corpora can be annotated to show anaphoric relations (coreference annotation), pragmatic information like speech acts (pragmatic annotation) or stylistic features such as speech and thought presentation (stylistic annotation). Of these the most widespread

type of annotation is POS tagging, which has been successfully applied to many languages; syntactic parsing is also developing rapidly while some types of annotation (e.g. discoursal and pragmatic annotations) are presently relatively undeveloped. In this unit we will introduce annotation types which are currently in general use by linguists. Readers are advised to refer to Garside, Leech and McEnery (1997: 85-90) and McEnery and Wilson (2001: 65-68) for discussions of annotation types not covered in this section.

4.4.1 POS tagging

POS tagging (also referred to as grammatical tagging or morpho-syntactic annotation) means assigning a part-of-speech mnemonic, also known as a POS tag, to each word in a corpus. POS tagging was one of the first widely used types of corpus annotation and is today by far the most common type. It is also the most basic type of corpus annotation forming the basis of further forms of analysis such as parsing and semantic annotation. However, corpora annotated for parts-of-speech alone are useful for a wide scope of applications ranging from disambiguating homographs to more sophisticated uses such as computing the occurrences of word classes in a corpus. Many linguistic analyses, e.g. the collocates of a word (see unit 10.2 for further discussion of collocation), also depend heavily on POS tagging (cf. Hunston 2002: 81).

Given the advanced state of the development of POS tagging, it can be performed automatically for many languages with a precision rate good enough for most research questions. The annotation tool which automatically assigns POS tags to lexical units is called a tagger. One of the best-known and most reliable taggers for English is CLAWS (Constituent-Likelihood Automatic Word Tagging System), developed at Lancaster University (see Garside, Leech and Sampson 1987). The system employs a hybrid statistical approach enhanced by a rule-based component, idiosyncratically called the 'idiom list' (Garside and Smith 1997). This tagger is reported to have achieved an accuracy rate of 97% on general written English (cf. Garside and Smith 1997). The system was employed to tag the British National Corpus (BNC, see unit 7.2). POS taggers have also been developed successfully for languages such as French (Gendner 2002), Spanish (Farwell, Helmreich and Casper 1995), German (Hinrichs, Kübler, Müller and Ule 2002), Swedish (Cutting 1994) and Chinese (Chang and Chen 1993; Zhang and Liu 2002).

The POS tags used by taggers such as CLAWS can be stored in different encoding formats. For example, a POS tag can be joined with a lexical unit by the underscore as in *going_VVGK*, using TEI entity references (see unit 3.3) as in *going&VVGK;*, using SGML (*<w POS=VVGK>going</w>*, or a simplified SGML form *<w VVGK>going* as in the BNC), or in XML format (*<w POS="VVGK">going</w>*). Whatever encoding style is adopted, these formats can be translated readily between each other to meet the need of individual corpus exploration tools. For relatively non-markup aware concordancers (e.g. MonoConc and WordSmith), the embedded annotation format using the underscore character may be preferred while for fully markup aware tools like SARA and Xaira, an SGML or XML format is preferable.

In part-of-speech annotation, the first issue that an annotator must address is how to segment the text into word tokens. The process of defining legitimate words in a running text is referred to as *word segmentation* or *tokenization*. For alphabetical languages like English, word tokens in a written text are normally delimited by a preceding and following space or new-line character. The one-to-one correspondence

between orthographic and morpho-syntactic word tokens can be considered as a default with three main exceptions: multiwords (e.g. *so that* and *in spite of*), mergers (e.g. *can't* and *gonna*) and variably spelt compounds (e.g. *noticeboard*, *notice-board*, *notice board*). To give an example of how a system may treat such cases, CLAWS treats a multiword expression as a single lexical unit and assigns so-called ditto tags to the whole expression. Ditto tagging involves assigning the same POS code to each word in an idiomatic expression, marking each with a two-digit number, the first digit of which indicates the total number of words in the expression while the second indicates the number of the word in the expression. For example, the subordinating conjunction (CS) *so that* is tagged as *so_CS21 that_CS22* while the preposition (II) *in spite of* is tagged as *in_II31 spite_II32 of_II33*. In contrast, the sub-parts of mergers are considered to be separate words. For example, the contracted negated modal verb *can't* is broken apart and tagged as *ca_VM n't_XX* while the catenative *gonna* is broken apart and tagged as *gon_VVGK na_TO*. A compound is tagged as one morpho-syntactic word if it appears as one orthographic word with or without a hyphen (*noticeboard_NN1* and *notice-board_NN1*), or as two morpho-syntactic words if it appears as two orthographic words (*notice_NN1 board_NN1*).

For many other languages, word segmentation is a much more challenging task. In Chinese, for example, a written text contains a running string of characters with no delimiting spaces. Consequently, word segmentation in Chinese requires complex computer processing, which usually involves lexicon matching and the use of a statistical model. In this book we will not discuss the technical details of how segmentation is achieved by programs working on Chinese. Readers interested in this should refer to Wu and Fung (1994), Huang et al (1997), Sun, Shen and Tsou (1998), Swen and Yu (1999), Feng (2001) and Zhang and Liu (2002).

4.4.2 Lemmatization

Lemmatization is a type of annotation that reduces the inflectional variants of words to their respective lexemes (or lemmas) as they appear in dictionary entries. For example, the lemma of *do*, *does*, *did*, *done* and *doing* is DO while the lemma for *corpus*, *corpora* and *corpuses* is CORPUS. Note that in corpus linguistics, as in this book, lemmas are conventionally written in small capital letters. Lemmatization is important in vocabulary studies and lexicography, e.g. in studying the distribution pattern of lexemes and improving dictionaries and computer lexicons (cf. McEnery and Wilson 2001: 53; Leech 1997a: 15).

Lemmatization can automatically be performed quite reliably for many languages including, for example, English, French and Spanish (cf. McEnery, Wilson, Sanchez-Leon and Nieto-Serano 1997). However, the usefulness of lemmatization depends on how inflectional a language is. For highly inflectional language like Russian and Spanish, where a lemma covers a large number of inflectional variants, lemmatization is particularly useful whereas for non-inflectional languages like Chinese, lemmatization is of limited use. As English is a language with simple inflectional morphology, which only inflects verbs for tense and nouns for plurality, lemmatization 'may be considered somewhat redundant' for English (Leech 1997a: 15). That may explain why, although quite accurate software is available for this purpose, few English corpora are lemmatized.

4.4.3 Parsing

As noted in unit 4.4.1, POS tagging is a basic step that often leads to further types of annotation such as parsing. Once a corpus is POS tagged, it is possible to bring these morpho-syntactic categories into higher-level syntactic relationships with one another (cf. McEnery and Wilson 2001: 53), in other words, to analyze the sentences in a corpus into their constituents. This procedure is referred to as *parsing*. As parsing often involves assigning phrase markers to constituents using sets of labelled brackets, parsing is sometimes referred to as *bracketing*, though strictly speaking, bracketing is specifically related to the labelling of phrase structures (PS grammar) while syntactic analysis may also cover dependency relations (constraint grammar) between words and functional labelling of elements such as subjects and objects etc (e.g. Karlsson, Voutilainen, Heikkilä and Anttila 1995). As parsed corpora, especially those in a vertical indented layout (as used in the Penn Treebank, see Marcus, Santorini and Marcinkiewicz 1993), often appear similar to tree diagrams, they are sometimes known as *treebanks*. For example, *Mary visited a very nice boy* can be bracketed as follows (cited from Santorini 1991: 3):

```
(S      (NP     Mary)
        (VP     visited)
                (NP     a
                        (ADJP very nice)
                        boy)))
```

Here, *S* represents *sentence* while *NP*, *VP* and *ADJP* stand respectively for *noun*, *verb* and *adjectival phrases*.

Parsing is probably the most common type of annotation after POS tagging. It is important to most natural language processing (NLP) applications – to make sense of a natural language, an NLP system must be able to decode its syntax. Syntactically parsed treebanks are even more useful than POS tagged corpora in linguistic research, as they not only provide part-of-speech information for individual words but also indicate constituent types and membership. For example, it is much easier to study clause types using a parsed corpus. A carefully edited treebank can also be used as a grammar tutor to teach grammatical analysis to students (cf. McEnery, Baker and Hutchinson 1997).

While parsing can be automated, its precision rate is generally much lower than that of POS tagging (cf. Mitkov 2002: 194). Typically an automatically parsed corpus needs to be corrected by hand (e.g. the Penn Treebank). Some available treebanks are entirely hand crafted (e.g. the Lancaster-Leeds Treebank), though interactive computer programs (e.g. EPICS, see Garside 1993) can be used to assist in the process of manual parsing. A hand crafted or post-edited treebank can in turn be used to train an automatic parser.

Syntactic parsing, no matter whether it is automatic or manual, is typically based on some form of context free grammar, for example, phrase-structure grammar, dependency grammar or functional grammar. Even with the same grammar, however, different annotators may use different parsing schemes, which may differ, for example, in the number of constituent types and the rules for combining each other (cf. McEnery and Wilson 2001: 55). For example, whilst the UCREL parsing scheme and the Penn Treebank scheme both employ a phrase structure grammar and cover noun, verb, adjective, adverbial and preposition phrases, the former also distinguishes between different clause types such as adverbial clause, comparative clause, nominal

clause, and relative clause whereas the latter differentiates between different types of *wh*-clauses (e.g. noun, adverb and prepositional phrases).

In terms of the details encoded, parsing can be either full parsing or skeleton (shallow) parsing. Whilst the former provides a syntactic analysis which is as detailed as possible, the latter tends to use less fine-grained constituent types. The Penn Treebank is an example of skeleton parsing. Here, for example, all noun phrases are labelled as N, whereas full parsing would distinguish between types of noun phrases (e.g. singular vs. plural). While skeleton parsing cannot provide as much information as full parsing, it allows for human analysts to parse or post-edit a corpus more speedily and consistently. Since automatic parsing is not yet sufficiently reliable (cf. Mooney 2003: 388; Collins 1997), human parsing or post-editing cannot be completely dispensed with in the treebanking process.

4.4.4 Semantic annotation

Semantic annotation assigns codes indicating the semantic features or the semantic fields of the words in a text. There are actually at least two broad types of semantic annotation. The first type marks the semantic relationships between the constituents in a sentence (e.g. as happens in the Penn Treebank, see Kinsbury, Palmer and Marcus 2002) while the second type marks the semantic features of words in a text. The first type is also known as 'semantic parsing' (Mooney 2003: 389) and should, in our view, be considered as a syntactic level annotation. We will confine ourselves to the latter type of semantic annotation in this book as this is by far the more common type at the present time. Semantic annotation is also referred to as word sense tagging. Annotation of this type is particularly useful in content analysis. Thomas and Wilson (1996), for example, find on the basis of a semantically annotated corpus of doctor-patient discourse that patients are more satisfied when doctors use more interactive words (e.g. discourse particles, first and second person pronouns, downtoners and boosters; see case study 5).

Semantic annotation is a more challenging task than POS tagging and syntactic parsing, because it is principally knowledge-based (requiring ontologies and lexical resources like dictionaries and thesauri). A more statistically driven approach to the problem does seem possible, however (see Stevenson and Wilks 2003: 256-258 and Mooney 2003: 387 for a discussion of statistical and machine learning approaches to semantic annotation).

In spite of the challenging nature of automatic sense disambiguation, work on it has met with some success. For example, Rayson and Wilson (1996) report on USAS (UCREL Semantic Analysis System), which is designed to undertake the semantic analysis of present-day English. The USAS semantic tagset is composed of 21 major categories which are further divided into 232 sub-categories. The system first assigns a POS tag to each lexical unit (single word or idiomatic sequence) using the CLAWS tagger and then feeds the output into the semantic tagging suite called SEMTAG. Experiments with contemporary texts show that the system has a precision rate of about 92% (see Rayson 2001). In addition to USAS, efforts have been made elsewhere to automate semantic annotation, for example, Popov et al (2003), Wilks (2003) and Guthrie (2003).

4.4.5 Coreference annotation

Coreference annotation is a type of discourse level annotation which has been applied to a number of corpora. The major concern of coreference annotation is coreference

identification, e.g. the identification of coreferential relationships between pronouns and noun phrases. This type of annotation makes it possible to track how elements of a text are progressively interwoven so that cohesion is achieved, typically through the use of pronouns, repetition, substitution, ellipsis and so on. A simple example of anaphoric annotation is:

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

Here the number 6 is an index number while the less than character < indicates that a backward referential (anaphoric) link is present, i.e. *they* and *their* point backward to *the married couple* (cited from Garside, Fligelstone and Botley 1997: 68).

The annotation scheme used in the above example is the Lancaster/IBM scheme (see Fligelstone 1991, 1992), which is based on Halliday and Hasan (1976) and Quirk et al (1985). This scheme aims at 1) identifying an anaphor/cataphor and its antecedent/postcedent, or establishing whether it is identifiable; 2) identifying the direction of a referential link; 3) identifying the type of relationship (e.g. reference, substitution, ellipsis, etc); 4) categorizing antecedents/postcedents; and 5) indicating semantic/pragmatic features (e.g. singular vs. plural, primary vs. secondary reference, exclusive vs. inclusive of addressee(s)). The Lancaster/IBM scheme was used in annotating the so-called Lancaster/IBM anaphoric treebank, which contains 100,000 words.

In addition to the Lancaster/IBM anaphoric treebank, there are a number of coreference annotated corpora, including for example, a 65,000-word corpus resulting from the MUC (Message Understanding Conference) coreference task (Hirschman 1997), a 60,000-word corpus produced at the University of Wolverhampton (Mitkov et al 2000), and 93,931 words of the Penn Treebank (Ge 1998). A much larger corpus annotated for coreference (one million words) is under construction on a project undertaken by the University of Stendahl and Xerox Research Centre Europe (Tutin et al 2000).

There is so far no generally agreed upon scheme for coreference annotation (see Garside, Fligelstone and Botley 1997 for a comparison of available annotation schemes). While the Lancaster/IBM scheme proposed one approach to coreference annotation, other schemes have also been developed. For example, Botley and McEnery (2001) in work focusing solely on demonstratives propose a scheme which encodes a set of five distinctive features: recoverability of antecedent (R), direction of reference (D), phoric type (P), syntactic function (S) and antecedent type (A), with each feature constituting an unordered set consisting of values relating to different categories of demonstrative use. For example, the code DARMN represents a referential usage (R) where a nominal (N) antecedent precedes an anaphor (A) and is directly recoverable (D) while the anaphor functions syntactically as a modifier (M). A modified version of this scheme was used to analyze demonstratives in Hindi (see Baker et al 2004).

Another scheme is the SGML-compliant MUC scheme, which has been used by a number of researchers to annotate coreferential links (e.g. Gaizauskas and Humphreys 1996; Mitkov et al 1999). The basic MUC scheme identifies a coreferential link using a unique identity number. It also encodes the type of coreference, as shown in the following example (cited from Mitkov 2002: 134), in which *IDENT* indicates the identity relationship between anaphor and antecedent:

<COREF ID="100">The Kenya Wildlife Service</COREF> estimates

<COREF ID="101" TYPE=IDENT REF="100">it</COREF> loses $1.2 million a year in park entry fee because of fraud.

Until recently, hardly any fully automatic coreference annotation system was reported to have achieved an accuracy rate which would allow it to be used as an automated or semi-automatic system for coreference annotation (cf. Mitkov 2002: 169, 195). However, there are now a range of systems that facilitate coreference annotation (see Mitkov 2002 for a review). To give two examples, the first tool developed to support coreference annotation was called Xanadu, which was developed to apply the Lancaster/IBM scheme at Lancaster (see Garside 1993). A more recent tool, called CLinkA, was developed at Wolverhampton and operates by default on the MUC scheme, though the system also allows users to define their own annotation scheme (see Orasan 2000). As CLinkA supports Unicode, the tool is writing system independent and can be used to annotate languages written in a wide range of writing systems.

4.4.6 Pragmatic annotation

Pragmatic annotation is yet another type of annotation at the discourse level. At present the focus of pragmatic annotation appears to be on speech/dialogue acts in domain specific dialogue such as doctor-patient discourse and telephone conversations.

A number of dialogue based projects have been reported around the world, for example, the Edinburgh Map Task corpus (Anderson et al 1991), the TRAINS corpus developed at the University of Rochester (Allen et al 1996), the ATIS (Air Travel Information Service) project (Wang and Hirschberg 1992) and the German VERBMOBIL project (Alexandersson et al 1997). These dialogue systems are typically domain specific and task-driven. Since 1996 the Discourse Resource Initiative (DRI) has organized annual workshops in an attempt to unify previous and ongoing annotation work in dialogue coding. The DRI annotation scheme, known as DAMSL (Dialog Act Markup in Several Layers) specifies three layers of coding: segmentation (dividing dialogue into textual units – utterances), functional annotation (dialogue act annotation) and utterance tags (applying utterance tags that characterize the role of the utterance as a dialogue act). There are four major categories of utterance tags: 1) Communicative status (i.e. whether an utterance is intelligible and complete); 2) Information level and status (indicating the semantic content of the utterance and how it relates to the task in question); 3) Forward-looking communicative function (utterances that may constrain or affect the subsequent discourse, e.g. *assert*, *request*, *question* and *offer*); 4) Back-looking communicative function (utterances that relate to previous parts of the discourse, e.g. *accept*, *backchanneling* and *answer*). Of these, only the latter two communication function types (3 and 4) are directly relevant to an increasingly important form of pragmatic annotation – speech act annotation. Readers are advised to refer to Leech et al (2000) for further discussion of the representation and annotation of dialogue.

A good example of speech act annotation is provided by Leech and Weisser (2003). They produced an annotation scheme for their Speech Act Annotated Corpus (SPAAC) of telephone task-oriented dialogues. The SPAAC scheme consists of 41 speech-act categories including, for example, *accept*, *acknowledge*, *answer*, *confirm*, *correct*, *direct*, *echo*, *exclaim* and *greet*. In addition to the assignment of these initial speech-act categories, the system also supplies supplementary syntactic, semantic and pragmatic information for each speech act in terms of form (e.g. *declarative*, *yes-no*

question, *wh-question*, *imperative* and *fragment*), polarity (i.e. *positive* or *negative*), topic (e.g. *location*, *name*, *day*, *date*, *time* and *railcard* related to train journeys) and mode (e.g. semantic categories such as *deixis*, *probability* and *reason*).

Pragmatic annotation like this has not yet been fully automated. Nevertheless, human analysts can be assisted by computer programs in their annotation task. Weisser (2003), for example, has developed an efficient XML-compliant tool (SPAACy) to help human analysts to annotate speech acts semi-automatically. The MATE (Multi-Level Annotation Tools Engineering) project has also produced a flexible workbench which provides support for the annotation of speech and text (cf. Carletta and Isard 1999).

4.4.7 Stylistic annotation

While pragmatic annotation focuses on speech acts in dialogue, stylistic annotation is particularly associated with stylistic features in literary texts (cf. Leech, McEnery and Wynne 1997: 94; but see McIntyre et al 2003 for an account of annotating speech and thought in spoken data). An example of annotation of this latter type is the representation of people's speech and thoughts, known as speech and thought presentation (S&TP). In stylistics, there is a long tradition of focusing on the representation of speech and thought in written fiction (e.g. Leech and Short 1981). Yet the representation of speech and thoughts has long been of interest not only to stylisticians, but also to researchers in applied linguistics, philosophy and psychology (cf. McIntyre et al 2003: 513).

As far as we are aware, the only corpus which has been annotated for categories of speech and thought presentation is the Lancaster Speech, Thought and Writing Presentation Corpus (S&TP), which is composed of a written component and a spoken component, developed from 1994 to 2003. The written section, approximately 260,000 words in size, contains three narrative genres: prose fiction, newspaper reportage and (auto)biography, which are further divided into 'serious' and 'popular' sections (see Short et al 1999). The spoken section was created with the express aim of comparing S&TP in spoken and written languages systematically. It contains approximately 260,000 words, making it comparable in size to the ST&WP written section. The texts contained in the spoken corpus are drawn from two sources: 60 samples from the demographic section of the British National Corpus and 60 samples from oral history archives in the Centre for North West Regional Studies (CNWRS) at Lancaster University (see McIntyre et al 2003: 514). Both written and spoken components of the S&TP corpus are marked up using TEI-compliant SGML (see unit 3.3).

Given the differences between written and spoken data, the S&TP annotation schemes for the written and spoken components are slightly different. However, the main categories remain unchanged (see McIntyre et al 2003: 516 for an account of modifications). The main categories include the direct category (e.g. *direct speech*, *direct thought* and *direct writing*), the free direct category (e.g. *free direct speech*, *free direct thought* and *free direct writing*), the indirect category (e.g. *indirect speech*, *indirect thought* and *indirect writing*), the free indirect category (*free indirect speech*, *free indirect thought* and *free indirect writing*), the representation of speech/thought/writing act category, the representation of voice/internal state/writing category, and the report category (e.g. *report of speech*, *report of thought* and *report of writing*).

Because surface syntax cannot reliably indicate the stylistic features as outlined above, the automatic annotation of such categories is difficult. Unsurprisingly, therefore, the Lancaster S&TP corpus was annotated entirely by human analysts.

4.4.8 Error tagging

Error tagging is a special type of annotation which is specifically associated with learner corpora and geared toward language pedagogy (see units 7.8, 10.8 and 17.4). Annotation of this kind involves assigning codes indicating the types of errors occurring in a learner corpus (see case study 3 in Section C). Corpora annotated for learner errors can help to reveal the relative frequency of error types produced by learners of different L1 (first language) backgrounds and proficiency levels. They are also useful when exploring features of non-native language behaviour (e.g. overuse or under-use of certain linguistic features).

At present a number of error-tagged learner corpora are available. For example, the Cambridge Learner Corpus and the Longman Learners' Corpus, which cover a variety of L1 backgrounds, are partly tagged for learner errors. In addition, a number of error-tagged learner English corpora are available which cover only one L1 background, e.g. the Chinese Learner English Corpus (CLEC), the JEFLL (Japanese EFL Learner) corpus, the SST (Standard Speaking Test) corpus of Japanese learner English and part (round 100,000 words) of the HKUST Corpus of Learner English (see unit 7.8 for further details of these learner corpora).

Error tagging schemes vary to some extent from one corpus to another in terms of the number and types of error codes. However, most schemes have in common error types such as *omission*, *addition* and *misformation*. The Cambridge scheme, for example, includes six general error types: wrong word form used (F), something missing (M), word/phrase that needs replacing (R), unnecessary word/phrase (U) and word wrongly derived (D) (see Nicholls 2003: 573-574). The CLEC scheme consists of 61 error types clustered in 11 categories while the SST scheme contains 47 learner error types.

Error tagging is a laborious and time-consuming task as it is difficult to develop either rule-based or probabilistic programs to identify errors due in large part to a lack of information regarding error patterns and their frequencies with respect to learner groups (cf. Tono 2003: 804). In spite of this difficulty, a number of attempts have been made to automate part of the error tagging process. For example, Granger and her colleagues on the ICLE project have developed a Windows-based error editor (cf. Dagneaux, Denness and Granger 1998); Tono et al (2001) have also presented a generic error tagset and an associated error editor. In addition, some tools have also been developed for detecting specific types of errors, for example, missing articles (Mason and Uzar 2000).

4.4.9 Problem-oriented annotation

Most types of the annotation we have considered up to now are intended to be useful for a broad range of research questions, and attempt to employ widely agreed categories or analytical schemes. Yet not all annotation schemes have that goal, and the final type of annotation explored here, problem-oriented annotation, arguably does not. Problem-oriented annotation differs in two fundamental ways from most of the annotation types discussed above. First, it is not exhaustive – only the phenomenon directly relevant to a particular research question, rather than the entire contents of a corpus, is annotated. Second, the scheme for problem-oriented annotation is

developed not for its broad coverage and consensus-based theory-neutrality but for its relevance to the specific research question (cf. McEnery and Wilson 2001: 69). Problem-oriented annotation is similar to the notion of 'local grammar', developed by Gross (1993) and Hunston and Sinclair (2000). This kind of annotation is necessary and useful for research questions that cannot be addressed using currently available annotations. Using problem-oriented annotation, for instance, Gross (1993) describes ways of accounting for time expressions while Meyer and Tenny (1993) study apposition in English. More recently, Hunston (1999a) uses this kind of annotation to study how people talk about sameness and difference.

Problem-oriented annotation is entirely dependent on individual research questions and the resultant annotation schemes are typically idiosyncratic to the extent that they are consequently difficult to use to explore other research questions. Nevertheless, it is a very important annotation type to keep in mind when considering a corpus-based approach to a research question. Problem-oriented annotations, as is clear from the references above, are certainly a valuable way of gaining insights into a specific research question.

4.5 Embedded vs. stand-alone annotation

We have so far assumed that the process of annotation leads to information being mixed in the original corpus text or so-called base document when it is applied to a corpus (i.e. the annotation becomes so-called *embedded annotation*). However, the Corpus Encoding Standard (see unit 3.3) recommends the use of *stand-alone annotation*, whereby the annotation information is retained in separate SGML/XML documents (with different Document Type Definitions) and linked to the original and other annotation documents in hypertext format. Stand-alone annotation fully addresses one of the criticisms of corpus annotation, namely the wish to be able to view a corpus in its raw form even if it is annotated, though standalone annotation is not the only solution to this problem (see section 4.2). In contrast to embedded annotation, stand-alone annotation has a number of advantages (Ide 1998) as it:

- provides control over the distribution of base documents for legal purposes;
- enables annotation to be performed on base documents that cannot easily be altered (e.g. they are read-only);
- avoids the creation of potentially unwieldy documents;
- allows multiple overlapping hierarchies;
- allows for alternative annotation schemes to be applied to the same data (e.g. different POS tagsets);
- enables new annotation levels to be added without causing problems for existing levels of annotation or search tools;
- allows annotation at one level to be changed without affecting other levels.

Stand-alone annotation is in principle ideal and is certainly technically feasible. It may also represent the future standard for certain types of annotation. Presently, however, there are two problems associated with stand-alone annotation. The first issue is related to the complexity of corpus annotation. As can be seen from the previous sections, annotation may have multiple forms in a corpus. While some of these readily allow for the separation of annotation codes from base documents (e.g. lemmatization, POS tagging and semantic annotation), others may involve much more complexity in establishing links between codes and annotated items (e.g. coreference and stylistic annotations). Even if such links can be established, they are usually prone

to error. The second issue is purely practical. As far as we are aware, the currently available corpus exploration tools, including the latest versions of WordSmith (version 4, Scott 2003) and Xaira (Burnard and Todd 2003), have all been designed for use with embedded annotation. Stand-alone annotation, while appealing, is only useful when appropriate search tools are available for use on stand-alone annotated corpora.

4.6 Unit summary and looking ahead

This unit first discussed the rationale for corpus annotation in the context of addressing a number of criticisms of it, followed by a discussion of how annotation is achieved. The unit then moved on to introduce some important annotation types. Finally, we explored the advantages, and potential problems of standalone corpus annotation.

It is clear from this discussion that the type of annotation needed in a corpus is closely associated with the research question one seeks to address using the corpus. It is also important to note that some types of annotation, e.g. POS tagging, which can be automated reliably, have a broad range of uses and form the basis of some higher-level annotations like parsing.

This unit is a basic and non-technical introduction to corpus annotation. The exploration of the annotation types presented here is also far from exhaustive. Readers interested in the in-depth discussion of corpus annotation should refer to Garside, Leech and McEnery (1997) and Mitkov (2003). In addition to some annotation types which are particularly associated with monolingual data (e.g. phonetic/phonemic and prosodic annotations), we have omitted an important type of annotation linked to multilingual corpora from our discussion – alignment. As alignment is an important type of annotation used in building parallel corpora, alignment will be introduced in the next unit, where we will shift our focus to a discussion of multilingual corpora.