Technical Report

UCAM-CL-TR-779
ISSN 1476-2986

Number 779

UNIVERSITY OF CAMBRIDGE

Computer Laboratory

A text representation language for contextual and distributional processing

Eric K. Henderson

April 2010

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

http://www.cl.cam.ac.uk/

This technical report is based on a dissertation submitted 2009 by the author for the degree of Doctor of Philosophy to the University of Cambridge, Fitzwilliam College.

Technical reports published by the University of Cambridge Computer Laboratory are freely available via the Internet:

http://www.cl.cam.ac.uk/techreports/

ISSN 1476-2986

Abstract

This thesis examines distributional and contextual aspects of linguistic processing in relation to traditional symbolic approaches. Distributional processing is more commonly associated with statistical methods, while an integrated representation of context spanning document and syntactic structure is lacking in current linguistic representations. This thesis addresses both issues through a novel symbolic text representation language.

The text representation language encodes information from all levels of linguistic analysis in a semantically motivated form. Using object-oriented constructs in a recursive structure that can be derived from the syntactic parse, the language provides a common interface for symbolic and distributional processing. A key feature of the language is a recursive treatment of context at all levels of representation. The thesis gives a detailed account of the form and syntax of the language, as well as a treatment of several important constructions. Comparisons are made with other linguistic and semantic representations, and several of the distinguishing features are demonstrated through experiments.

The treatment of context in the representation language is discussed at length. The recursive structure employed in the representation is explained and motivated by issues involving document structure. Applications of the contextual representation in symbolic processing are demonstrated through several experiments.

Distributional processing is introduced using traditional statistical techniques to measure semantic similarity. Several extant similarity metrics are evaluated using a novel evaluation metric involving adjective antonyms. The results provide several insights into the nature of distributional processing, and this motivates a new approach based on characteristic adjectives.

Characteristic adjectives are distributionally derived and semantically differentiated vectors associated with a node in a semantic taxonomy. They are significantly lower-dimensioned then their undifferentiated source vectors, while retaining a strong correlation to their position in the semantic space. Their properties and derivation are

described in detail and an experimental evaluation of their semantic content is presented.

Finally, the distributional techniques to derive characteristic adjectives are extended to encompass symbolic processing. Rules involving several types of symbolic patterns are distributionally derived from a source corpus, and applied to the text representation language. Polysemy is addressed in the derivation by limiting distributional information to monosemous words. The derived rules show a significant improvement at disambiguating nouns in a test corpus.

Contents

1

Introduction

This thesis explores the integration of contextual and distributional information with symbolic information derived from linguistic analysis. I have three main objectives relating to this goal. First, to investigate representational issues in text processing by developing a text representation language which provides a novel framework for contextual and distributional processing and extends existing representational forms in several meaningful ways. Second, to develop a systematic treatment of structural and linguistic context in the representational language, based on the similarities between discourse segments and sentence phrasal structure. Third, to propose a method for integrating distributional information into the contextual representation and to explore novel methods of augmenting symbolic processing with distributional methods.

As an experimental framework, I will first develop a text representation language which integrates the representation of linguistic context with an object-oriented representation of syntactic dependency structure. This representation extends existing graph-based dependency structures with several features that facilitate contextual and distributional processing, allows rudimentary semantic capability, and supports direct internal manipulation to produce surface variations. A deterministic recursive algorithm is presented to realise the internal representation as a well-formed surface expression. I will give a formal description of the representation language, and a comprehensive account of the range of constructions it supports.

I will develop the notion of structural and linguistic context by examining discourse segments and phrase structure. I will propose a representational model of contexts that generalizes over both cases and integrates with the text representation. I will present several experiments that use symbolic processing and show that contextual information

(encoded in the representation) can be used to augment the symbolic processing and improve the results.

Next, I will introduce distributional processing through a novel evaluation of several existing distributional measures using adjective antonyms. This will suggest a novel semantic classification task using the distributional data derived from the adjectives. I will explore this in depth and present an experiment designed to test the properties of the distributional data.

Finally, I will consider the integration of distributional data with the symbolic representation language. By extending the previous distributional techniques I will present an experiment which derives symbolic rules for determining nominal sense information. The rules are applied to the representation language to decide broad semantic classification of nouns, which can be used to augment symbolic tasks.

1.1 A Text Representation

The integration of contextual and distributional information with other linguistic information is best achieved at the representational level. However, most text representation languages do not include document level structural information or corpus-based statistics because they operate at the level of sentences or words. (Phrasal context is also not typically encoded explicitly, although it can be recovered from the dependency structure in most cases).

A treatment of contextual and distributional information in a text representation, similar to other symbolic information, would allow text processing to take advantage of this information without requiring significant adaptation. An experimental framework which includes a text representation that supports the contextual and distributional extensions proposed in this thesis, as well as other representational properties, is integral to the work presented. In this section I will discuss the requirements of a text representation having certain properties that fit with the direction of the thesis. I will propose specific desiderata and introduce a representation that satisfies these requirements. This representation will be examined in detail in the next chapter, and used in the experiments in the remainder of the thesis.

1.1.1 Desiderata

All textual language processing begins with some type of analysis of the surface text. Even shallow approaches to statistical processing often employ morphological analysis and part-of-speech tagging. An important step in any textual analysis is transforming the surface text into a suitable representation because the raw surface text is not a very efficient form for computation. A computational representation encodes the text, using the results of the analysis, in a more tractable form. The actual syntax and structure of a representation will largely depend on the linguistic level of the task implementing the analysis. This can vary widely, depending on the type of task and technologies applied. At one end of the scale, shallow approaches utilize a minimum of linguistic analysis, and at the other end deep processing techniques rely on multiple levels of syntactic and semantic processing. The result is a corresponding range of representational forms, many of which are not necessarily compatible.

It has recently been suggested (de Marneffe and Manning, 2008), that some of these representational forms used within the linguistics community prove to be daunting to non-linguistic researchers when attempting to apply them to specialised domains. Specifically, de Marneffe and Manning (2008) suggest that many deep syntactic and semantic constructs which are typically encoded in linguistic representations, have little practical value in non-linguistic applications that require text processing. For example, the distinction between an argument and adjunct in a dependency representation may have little significance when mining text for nominal compounds. This deeper linguistic information is seen as a liability by de Marneffe and Manning (2008), because it unnecessarily obfuscates the representation for the non-linguistic user community. They argue for a simplified linguistic encoding, such as the Stanford typed dependency representation, which is accessible to non-linguists while retaining the salient information for useful text processing.

While a streamlined representation may be advantageous for many applications, it is also true that more sophisticated tasks require more linguistic information. For example, McConville and Dzikovska (2008) report on the linguistic information needed for a tutorial dialogue system. They evaluate five representational forms of labelled grammatical dependencies, focusing on four specific linguistic phenomena (passive,

control and raising, noun modifiers, and prepositional significance). The representations are interpreted as input to a semantic processor, and judged by their facility to provide the necessary linguistic cues for deriving a semantic representation. McConville and Dzikovska (2008) conclude that no single representation is satisfactory for their needs, although they find all the desired features within the set of representations.

The recent focus on representational issues demonstrates there is a need for investigating better representational approaches. On the one hand, deeper representations such as phrase structures can seem "much more foreign and forbidding" (de Marneffe and Manning, 2008) and limit their utility to the wider research community, while on the other hand, an information-poor representation will not be able to serve the needs of more sophisticated language processing tasks. One approach to addressing both of these levels is an intermediate representation with task-specific transformations. This is an active area of research and does not necessarily solve the fundamental question of the primary representational form, however part of the work presented in this thesis (CAMEO) addresses many aspects of linguistic constructions which must be covered by an intermediate representation.

Representational forms are often developed as a complement to a particular language technology, e.g. a parsing system. It is only recently that these systems have matured to the point that wide scale evaluations of competing technologies have become feasible. This in turn has prompted a push to find a superset of representational features that can serve as a normalized reference representation of the disparate linguistic output. For example, Flickinger (2008) presents several desiderata for labelled dependency annotation. Although the CAMEO representation is intended for more than strict annotation, it encompasses most, if not all, of the desiderata proposed by the parser evaluation community. Globally unique identifiers, a means to identify the root predication, and properties of entities and events, are a few examples of the desiderata proposed in Flickinger (2008) which are incorporated into the CAMEO representation. However, the capability for interpretation and realisation places further demands on the CAMEO representation, and because of this an extended set of desiderata is warranted.

When developing an experimental framework for text processing, certain properties of the internal representation are demanded by the particular task being undertaken. However, there are general properties of the representation to consider as well, such as

the elimination of redundant processing. The development of a representation of context addressed in this thesis requires an examination of the general form and functions inherent in a text representation language. The following desiderata will be used to guide the general design and evaluation of the text representation developed in the first part of the thesis. Each of these desiderata will be justified in turn below.

General :	*support a wide range of derivative representations*
Rich :	*capture a maximum set of surface information*
Reversible :	*support systematic realisation of surface text*
Incremental/Robust :	*support all levels and stages of linguistic analyses*
Precise/Unambiguous :	*support efficient equivalency testing*
Flexible :	*allow easy manipulation of internal representation*
Primitive Semantics :	*support a minimal meaning capability*

1.1.1.1 General

A common representation can prevent the need for redundant low-level analyses when working with multiple processors that use different technologies. For example, semantic information can sometimes be derived from a shallower syntactic representation (Johansson and Nugues, 2008). However, a semantic transformation based on a specific syntactic representation makes it non-trivial to substitute a different syntactic processor. Using a generalized intermediate representation can enable transformations from various syntactic processors to various semantic representations, via the intermediate representation.

Another benefit of a generalized intermediate representation is the facilitation of evaluation between competing technologies. Output from different technologies can be normalized to the intermediate representation for comparison, making objective evaluations more feasible (Srinivas et al. 1996, Carroll et al. 1998, Flickinger 2008, Tateisi 2008).

Once deeper linguistic and semantic processing takes place, representational forms tend to diverge further. Part of this is a natural result of the theory governing the processing, but this also results from independent development of redundant functions.

A similar challenge is faced by the annotation community. There exists several styles of annotation which denote similar linguistic items, for example TreeBank, BNC, and Brown use different variants of POS tags. Semantic annotation in the FrameNet, VerbNet, TimeBank, and NomBank corpora may denote different semantic concepts, but also have different formats and structures.

The annotation community has proposed several solutions to this problem ranging from successive transformations (Hajičová and Kučerová, 2002), to a schematic syntactic representation that can be combined with idiosyncratic lexicons to derive theory, domain, or application specific representations (Pajas and Štěpánek, 2008). In addition, work on standardizing many aspects of linguistic representation is currently underway. For example, the International Organization for Standardization (ISO) is developing standards for lexical (LAF), morphosyntactic (MAF), and lexical resource (LMF) annotation. These schemes are meta-level representations designed for annotation of a wide range of linguistic phenomena at various detail and levels. They generally have a much wider scope than representations used in text processing, since they often include meta-linguistic annotation. (Because the focus of this thesis is on techniques for representations used in processing and not general annotation issues, I will not review the various proposed annotation standards.)

Many representations used in text processing are derived from a specific linguistic theory. Ideally a theory-neutral representation would have the widest utility; however this can be problematic for some levels of syntactic representation. For example, some syntactic theories encode long-distance dependencies and others do not. Unifying representations of deeper semantic representations may pose even more problems, although there are current efforts to pursue this approach (Pustejovsky et al., 2005).

1.1.1.2 Reversible

Certain text processing tasks produce natural language as output, for example QA and text summarization. A text representation should therefore be reversible, allowing for realisation of well-formed surface text from the internal representation. This allows

modules that do not require sophisticated language generation planning to operate directly on the representation and produce valid textual output.

The representation should be linguistically rich enough that the surface realisation can be affected through a deterministic transformation which does not require a grammar. This decouples the representation from specific grammars and allows it to remain theory-agnostic.

1.1.1.3 Incremental / Robust

In order to support a wide range of linguistic processing, the representation must also be robust and compositional. The representation should serve as both the input and output stages to a wide range of processors so the greatest reuse is achieved. Additionally, the representation should have a strategy to support shallow processors that do incremental analysis, followed by deeper processing that takes advantage of the incremental analysis without requiring explicit compatibility.

1.1.1.4 Precise / Unambiguous

One common function of a representational language is testing equivalency. Linguistic tasks that rely on some measure of syntactic or semantic distance usually require some means to compare sentences or sections of text. Thus the representation must be precise, unambiguous, and allow comparison of representational forms.

1.1.1.5 Flexible

It is also important that the representation be flexible so that manipulating a representation programmatically (or by hand) is manageable. This is especially important in an experimental framework where it is sometimes necessary to hand-correct the outputs of a given stage for the purpose of testing a hypothesis. For example, Levy and Andrew (2006) note there is often a need for tree manipulation in the development and use of annotated corpora. They describe a specialized system for manipulating representations that use syntactic tree structures. A general intermediate representation should not require special tools for manipulation. If a representation

includes complex sentence structures which are unintuitive and difficult to edit, its utility is diminished.

1.1.1.6 Primitive Semantics

Although a general intermediate representation must be theory neutral, it should have some minimal semantic properties. Allowing for rudimentary meaning representations similar to those found in a knowledge base gives the representation a wider utility, especially for derivative semantic representations. Support for assertional statements that do not require complete syntactic dependencies and are distinguishable from syntactic analyses can provide primitive semantic capability for tasks that do not require deep semantic processing.

1.1.2 Deficiencies in Existing Representations

In Chapter 2 I will survey some of the existing representations which appear in the literature. However, to help make clear the motivation for developing a new text representation language, I will mention some of the deficiencies of existing representations here.

Annotation schemes are not technically representation languages, but they share common representational forms. There are numerous XML-based annotation schemes and many approaches to generalize them. However, because they are intended as adjunct to surface text, they do not have an internal representation. This precludes them from being used to do shallow semantic processing or any form of internal manipulation. For example, it would not be possible to instantiate objects which represent semantic individuals yet have no determined surface realisation, as a rudimentary semantic capability supports.

Syntactically-based representations, such as Grammatical Relations (GRs), are better suited for analysis and lack features to support generative operations such as surface realisation. The declarative grammars that produce these representations can usually be run in reverse to produce surface text, but this is not always possible from the

representational form. Deeper semantic representations, such as QLF (see Section 2.2.5), will often include a generation component. However, this typically operates at a more abstract level and is usually non-deterministic, requiring a theory specific grammar.

Internal manipulation of the representation, though theoretically possible in any representation, is not specifically addressed in many representations. Semantic representations such as QLF and RMRS, which link certain types of semantic objects, would be most amenable to this type of operation. However, manipulating the shallower syntactic representations would amount to simply rewriting surface text, which would require linguistic knowledge of syntax and grammar.

1.1.3 The CAMEO Representation Language

The experimental text representation language developed in this thesis to investigate the stated desiderata is called CAMEO. It defines a set of elements and attributes and the rules for transforming them to/from surface text. The fundamental elements of the language are *events* and *objects*, corresponding roughly to verbs and nouns. There is no semantic significance to either type except as pertains to its role in the representation. (The goal here is simply to produce an intermediate representation that can be manipulated and possibly transformed into other higher-level representations.)

Another important feature of the CAMEO representation, which is necessary for the goals of this research, is the abstraction of contexts. A context is modelled as a scope or container and it is used to denote several types of textual segmentation including document level (e.g. *chapter*, *paragraph*, *sentence*) and phrase level (e.g. *clause*, *quotes*, *citations*).

Besides events and objects there are several other primitive elements defined by the language as well as attributes which attach to them. These are detailed in a further chapter, but for illustrative purposes I will give a brief example.

Consider the sentence shown in Figure 1.3.3, which appears, say, in a fictitious document in the twelfth paragraph. The representation of this example in the language is given below.

[*I have property in Manilla*]

```
ctx [ ID = 1  TYPE = doc  TITLE = Living Abroad  AUTHOR = Jane Smith
    ctx [ ID = 12 TYPE = par
        ctx [ ID = 13 TYPE = clause

            obj[ ID = 567  PRON = I  ]
            obj[ ID = 568  CLASS = property  ]
            obj[ ID = 569  name[ Manilla ]]

            evt [ ID = 231  ACTION  = have
                  SUBJECT = obj(567)
                  OBJECT  = obj(568)
                  rel[ PREP = in  OBJECT = obj(569) ]
            ]
        ]
    ]
]
```

Figure 1.1.3 – Example framework representation

The notation is simplified for clarity and the entire document would of course contain many more contexts and objects. From this it is possible to see the general form of the representation. Containers (scopes) are delimited using **[ ]**, and each container is assigned a unique id (in the examples throughout this thesis, some of these are omitted for readability).

The opening context (**id**(1)) represents the entire document, in this case *Living Abroad*. Since there are no formal chapters in this document there are no contexts of type *chapter*. Each new paragraph can be represented as a new context of type *par*, which in turn contain contexts of type *clause*.

Within the clause context, there is an object (**obj**) representing each noun of the clause. The objects participate in the event (**evt**), which represents the verb. This sentence also contains a prepositional phrase (see **rel** in Figure 1.1.3).

Notice the three nominals in this sentence all have the same general representation, even though one is a pronoun and one is a proper noun. This is an example of the rudimentary nature of the semantics included in the representation. Further semantic analysis on these nouns would require a separate module be implemented that operates on the representation. However, even at this low level some shallow semantic processing tasks are possible.

It should also be noted that not all the attributes and possible contexts will necessarily be discovered, depending on the document and the initial processing. For instance, some documents may have structural mark-up indicating titles, by-lines, etc. These attributes are easily transferred into the representation using simple transformational rules. On the other hand, when this kind of information occurs as raw text, i.e. with no mark-up, special processing beyond the standard syntactic parsing becomes necessary to recover their special status.

1.1.4 Objectives of the Thesis

The main objectives of the thesis were explained at the beginning of this chapter. I will summarize these objectives below along with the expected outcomes and enumerate the evaluation criteria that will be used to determine the success or failure.

- **Develop a text representation language which satisfies the desiderata**

The evaluation criteria for this objective will be the desiderata proposed earlier in the chapter. Part of the success will be determined by successfully implementing and using the representation in subsequent experiments in the thesis. However, several specific evaluations of aspects of the representation will be performed:

1. Deriving CAMEO representations from two separate syntactic processors (RASP and Link Grammar), allowing for hand-annotation of unsupported analyses (e.g. contexts).

2. Transforming the CAMEO representations into application-specific representations and annotations.

3. Recovering the surface text systematically from the internal CAMEO representation.

4. Demonstrating surface manipulation through the internal representation

- **Develop a systematic treatment of structural and linguistic context.**

Success of this objective will be measured using several representative task-level evaluations. The contribution of context, and the contextual model of representation, will be evaluated on several aspects of anaphora resolution and judged successful if some level of improvement can be achieved.

- **Explore distributional methods generally and develop a specific novel application to symbolic processing**

An evaluative experiment designed to measure the effect of distributional information integrated with the symbolic representation will be used to determine the success of this objective.

1.1.5 Outline of Remaining Chapters

- **Chapter 2** –A critical review of the literature is presented. Relevant work on incorporating context is examined. Recent attempts at combining symbolic and distributional processing are noted. A comparison of several comparable text representations is given with the proposed CAMEO extensions contrasted.
- **Chapter 3** –The CAMEO representation language is described in detail. The fundamental elements of the representation are explained, followed by a detailed account of notable constructions. A formal definition of the language syntax is given, along with a description of the processing required to produce the representation from a specific syntactic analysis.
- **Chapter 4** – Two important operations on the representation are explored. In the first half of the chapter, a treatment of the surface transformation is presented including evaluative experiments which test the range of surface expressions supported by the transformation. The second half of the chapter explores applications of text manipulation, giving qualitative arguments as well as a quantitative experiment to evaluate the manipulative capabilities of the representation.
- **Chapter 5** –An analysis of context at the structural and syntactic level is presented. A general recursive representation of context is proposed and demonstrated. The application of contextual information to reference resolution is evaluated through several task-level experiments.
- **Chapter 6** – The integration of symbolic and distributional information in the text representation language is discussed. The internal support for distributional processing is explained, along with the methods used for annotating the symbolic representation with information derived from external sources.

- **Chapter 7** – Distributional methods are introduced by examining statistical processing techniques for lexical semantic acquisition. Several semantic similarity measures and an experiment designed to evaluate these measures using adjectives are described.
- **Chapter 8** – Methods of combining distributional and symbolic processing are investigated. A novel approach to measuring the semantic similarity of nouns, based on the lexical properties of uniquely differentiated adjectives (referred to as *characteristic adjectives*), is developed using the distributional data from the previous chapter, combined with symbolic lexical information. Experiments meant to test the performance of this similarity measure are presented.
- **Chapter 9** – The distributional process of deriving characteristic adjectives is extended to encompass shallow symbolic dependency information for verbs and nouns. The result is a list of distributionally derived symbolic rules, which give a strong indication of a noun's semantic class. The rules are applied to the text representation and an evaluation of their performance is reported.
- **Chapter 10 --** The main points of the thesis are summarized, along with conclusions and some comments on future work.

2

Literature Review

2.1 Context

There are two aspects of context considered in this research: document structure, and syntactic context. Document structure has been used more prominently than syntactic context in the IR community. For example, Kazai et al. (2001) propose a hierarchical representation of structured documents for IR. This representation is used in a model of weighting by recursing through the document structure, aggregating the weights of child nodes with that of its parent. The structural representation of the document was also used to determine the best point of entry to the document (in the case of a web search referral).

Documents that do not have explicitly marked structure can still benefit from these techniques if the structure can be induced. For instance, Nomoto and Matsumoto (1996) use automatically acquired text structure to improve topic identification. Using a measure of similarity between paragraphs and the article title, they remove irrelevant passages to induce paragraph structure, resulting in improved performance on automatic topic identification.

In the NLP community it appears there has been less attention paid to document structure. Goecke and Witt (2006) posit that document structure provides an important context for anaphora resolution. Their corpus study reveals that there are anaphoric references that span distances that appear long when considered linearly, but reasonable when document structure and context is considered. For example, a discourse referent might be introduced followed by a list of items, or perhaps, a quotation. It is not

unreasonable to then have an anaphoric reference to this referent even though many sentences (the list) intervene. Taking into account the document structure and the fact that the text following the discourse referent was a list is hypothesized to help resolve some of these references.

The consideration of the *global* document context (versus structure) has been successfully applied to several NLP tasks over the past several years. Mikheev (2000) considers the global document context when resolving sentence boundaries, capitalization, and abbreviations. Gale, Church and Yarowsky (1992) show that words tend to exhibit a single sense within a global document context, and Yarowsky (1995) shows an improvement on a word sense disambiguation task when global document context is considered.

In addition to the document surface structure, the content of a document has been hypothesized to exhibit a discourse structure. Discourse structure has been an active area of research in the NLP community, and different theories of discourse coherence have been proposed including Hobbs (1985), Grosz and Sidner (1986), Mann and Thompson (1987), and McKeown (1985). Applications of discourse structure include topic and sentiment identification, summarization, generation, and simplification. Because discourse structure is arguably a semantic phenomenon and not necessarily syntactic, it is outside the scope of this thesis.

2.2 Existing Representation Strategies

Representations, whether linguistic or semantic, are often described in terms of a formal theory and are themselves sometimes incompletely formalized. Literature on representational languages highlights the aspects that differentiate the language, and may leave out the details on other constructions. A comprehensive comparison then becomes difficult. However, all representations ultimately derive from the surface string and thus can be expected to carry some amount of similar information. For example, word stems and morphological information, verb tense and mood, verbal argument structure, and nominal number are typical features that can be recovered from a representation. In this section I will survey several existing representations that I will use as a contrast to CAMEO, the representation developed in this thesis. All of the

representations examined here (including CAMEO) are ultimately dependency- and graph-based (at some level). The differences arise with how the representations extend the basic dependency information to enable deeper syntactic or semantic processing.

2.2.1 Annotation Schemes

The annotation community has produced several schemes which, though technically not representations, serve related purposes. As the demand for larger corpora with more sophisticated annotation grows, these schemes have been adapted to cover some of the same issues facing a surface representation (e.g. long-term dependencies).

Annotation schemes initially were developed independently and in a task-dependent manner. This produced syntactic annotations such as TreeBank, BNC (2002), and the Brown Corpus, and very different semantic annotations, such as FrameNet (Ruppenhofer, 2005), VerbNet (Schuler, 2005), TimeBank (Setzer, 2001), and NomBank (Meyers et al., 2004). The current trend is to augment syntactic annotation with semantic information (Sgall et al., 2004) in schemes such as PropBank (Kingsbury and Palmer, 2002), LCS (Dorr, 2001), and PDT (Hajić , 1998).

Like intermediate representations, the stated goal of the annotation community is a theory-neutral scheme which has wide utility, but this has proven difficult even for some levels of syntactic features (e.g. HPSG vs. Dependency Grammars, long-distance dependencies, etc.). Attempts at merging existing schemes toward this goal are currently on-going (Pustojevsky et al., 2005).

The NITE Object Model (Carletta et al., 2003) is a language for developing multimodal annotation which uses a typed, object-based structure. Objects are nodes that participate in a graph structure, and can be specialized using attributes and features. Arbitrary annotation can be represented as long as it is derived and defined using the primitive object types defined in the system. Multiple sets of annotation are integrated via standoff notation that point to the data set or other annotation. This allows correlation of e.g. syntax, prosody, words, and gestures. The NITE system provides libraries for inspecting and manipulating the annotation sets.

2.2.2 Grammatical Relations

Grammatical relations (GRs) are a linguistic representation that encodes local head/dependency relations. They are typically expressed as a list of tuples that consist of the name of the relation, the lexical head, and any lexical daughters. Using GRs in parser evaluation was proposed by Lin (1995). Carroll et al. (1998) further refine this idea by proposing an annotation scheme designed to be an independent common language for evaluation and comparison of different parsing technologies. After suggesting that such a language loosely based on the feature structures of Lexical Functional Grammar (LFG) could support a comprehensive and unambiguous representation, the authors settle for a simpler, more practical approach using tuples (as previously described). Figure 2.2.2 shows an example GR representation and its corresponding parse tree (taken from Briscoe, 2002). This representation serves as a sort of "lowest common denominator" among different GR annotations. Both Carroll et al. (1998) and Yeh (2000) discuss the issues that arise from disparate GR representations.

```
(ncsubj leave:6_VV0 group:5_NN1 _)
(detmod _ debt+s:2_NN2 What:1_DDQ)
(ncmod _ group:5_NN1 Qintex:4_NP1)
(aux _ leave:6_VV0 do+ed:3_VDD)
(dobj leave:6_VV0 debt+s:2_NN2 _)
```

```
(T/txt-sc1/-+-
(S/whnp_s (NP/det_n What_DDQ
        (N1/n debt+s_NN2))
   (S/sai/- do+ed_VDD
     (S/np_vp
       (NP/name_n1
         (NP/name/-
         (N1/n Qintex_NP1))
           (N1/n group_NN1))
       (V/0 leave_VV0))))
?_?)
```

Figure 2.2.2 – Sample GR output (left) and corresponding parse tree (right)

GRs (as specified by Carroll et al., 1998) can be characterized as syntactic descriptions analogous to the functional descriptions of LFG. That is, they are a list of the syntactic dependencies encountered in the text, in the form of binary tuples which describe relationships in the dependency structure. It is not necessary to have a complete parse to produce GRs, and many partial parsing and alternative approaches

have been applied towards extracting them (e.g. Argamon et al. 1999; Grefenstette, 1999; Srinivas, 2000). GRs essentially comprise atomic relations that can be interpreted independently, and thus are suited for robust analysis that can support incremental composition (cf. RMRS), and they have been used increasingly in NLP research as an intermediate representation for applications beyond parser evaluation (e.g. Grefenstette, 1997; Palmer et al., 1993; Yeh, 2000).

However, there are some limitations to using GRs as an intermediate representation. Because GRs are a strictly lexical representation, they do not normally mark semantic interpretations. Also, depending on the annotation scheme and targeted application, some of the surface information relating to the phrase structure may be discarded by the representation. For some tasks it can be advantageous to have the surface syntactic features explicitly marked, so a more expressive representation is sometimes necessary.

Also, because the primary structure is a relation, lexemes can appear in multiple relational structures. This can make annotating GRs with lexical distributional information and/or using GRs to compile distributional information cumbersome.

Although it is not a typical application of GRs, direct manipulation using GRs would be less efficient than more structured representations because there is no explicit link to each relation a lexeme participates in, and vice versa. For example, changing the subject in a sentence would require searching each relation for an instance of the existing lexeme.

The CAMEO representation developed in this thesis is similar in some respects to GRs, but includes a few innovations to address some of these deficiencies. It is structurally more akin to the AVM model briefly suggested in Carroll et al. (1998), than their more commonly-used tuple list notation. More importantly, from this common fundamental representation, CAMEO has been extended to include a treatment of context at various linguistic levels.

It should be noted that like CAMEO, GRs are derived from a syntactic analysis of the surface text (usually a parse structure). Therefore, it is possible to use a GR representation as an alternative source of linguistic analysis for creating a CAMEO representation. As I will demonstrate, there is no requirement in the CAMEO

representation for a complete syntactic parse tree, so the possible fragmented nature of a GR list does not pose a problem. A more likely application however, would be to derive GRs from an existing CAMEO representation, which can be transformed easily into tuple-based GR annotation.

2.2.3 Lexical Functional Grammar

LFG is a theory and framework for representing both syntax and grammatical function (Kaplan and Bresnan, 1982). LFG includes two levels of syntactic representation: the c-structure, which is a standard tree representation of the constituent phrase structure, and the f-structure, which is an explicit description of the grammatical functions derived from the c-structure. The representation of f-structures takes the form of an AVM, i.e. attribute-value feature matrices in recursive structures. It is possible to reconstruct the c-structure from the f-structure, but to construct the f-structure from the c-structure requires a mapping function. Figure 2.2.3 shows an example sentence with the corresponding c-structure and f-structure (Dalyrmple 97).

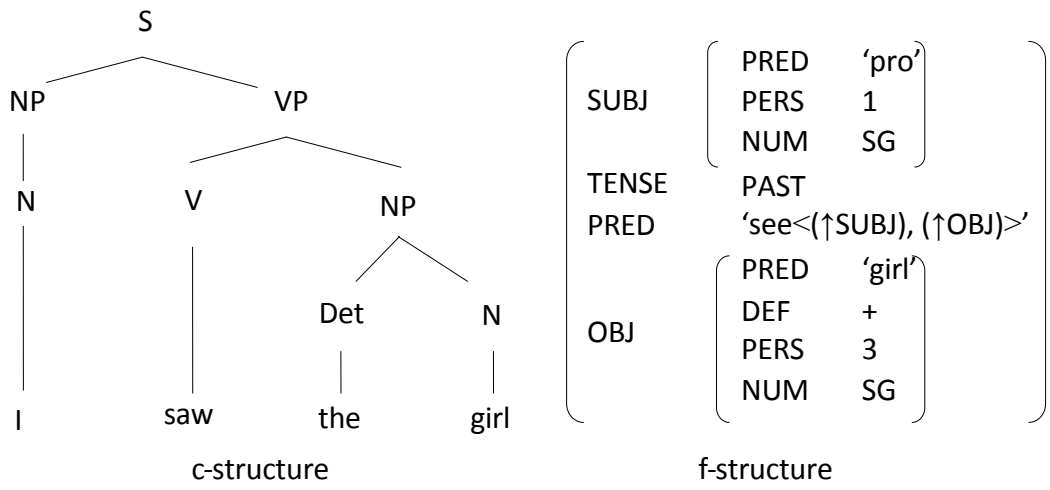


Figure 2.2.3 – Sample LFG representation for 'I saw the girl'

As a representational language, the LFG f-structure does well encoding surface features and syntactic relations. For example, in Figure 2.2.3, the direct object OBJ has encoded the DEF (definite) attribute, the person, and the number. However, because LFG is tied to a particular theory, it is less flexible in the syntactic interpretations it can represent. A less structured analysis, such as that produced by some dependency

grammars, might be problematic to represent in LFG since functional information may exist indirectly (Schneider, 1998). Also, LFG includes quasi-semantic information in certain functional control schema/structures which may not be present in the constituent structure. This limits its use as a neutral intermediate surface representation (which is not its intended purpose).

In contrast, CAMEO is not a syntactic or functional theory and does not include unifying features that constrain the syntactic analysis. It supports both linear and hierarchical representation strategies, allowing it to remain theory neutral and agnostic to the syntactic structure, as well as robust in the case of failed analyses.

Another problem which would arise from adopting LFG as an intermediate representation is its minimal support for local identifiers on certain recursive sub-structures, which are produced through multiple applications of a single schema (i.e. mapping functions). This reduces the flexibility for searching and manipulating document level representations. CAMEO uses identifiers which are integrated at all levels of the representation, treating all constituents as objects which are globally uniquely identified and easily manipulated.

In LFG, the representation of long distance dependencies evolved by allowing descriptive grammars (i.e. regular expressions) in the functional mappings from c-structures to f-structures (Kaplan and Zaenen, 1989). This allowed a finite description (required by the functional mapping constraints) of the potentially infinite number of mapping rules needed to cover the possibilities introduced by long distance dependencies. Although this innovation admitted a treatment of gap binding into the f-structure representation, it appears limited to intra-sentential linkage. CAMEO has the flexibility to model any gap binding in the surface syntactic analysis, regardless of distance, and also allows for direct linkage of any other binding information, e.g. from external analyzers.

Finally, although LFG includes some functional constraints on word order (amounting to f-structure precedence), this does not extend to the attribute level. CAMEO includes much finer control over surface word order, through the use of explicit attributes which can indicate the position of certain function words.

2.2.4 First Order Predicate Calculus

First Order Predicate Calculus (FOPC) is a semantic representation language that has been used widely in the research community. It is a flexible, relatively simple method for encoding meaning in logical forms. This makes it well-suited for assertional databases that represent model-theoretic knowledge. Thus, its use as a compositional semantic representation allows for a single representation in both cases.

There are many different forms of FOPC, but it is essentially comprised of **terms**, which represent objects (or object collections), **relations** among the various terms (e.g. verb events), and **logical connectives** (e.g. conjunction $\wedge$, disjunction $\vee$, and negation $\neg$). Beyond these basic components of FOPC, there are also two operators that are used to encode quantification: the existential quantifier $\exists$, and the universal quantifier $\forall$.

The basic FOPC framework has been used in developing strategies to represent natural language constructs by linguists such as Montague (1973), Davidson (1967), Parsons (1990), and others. Davidson proposed a reification of events that allows variable arity relations. For example, the sentences in (1) all have a different number of verbal arguments:

(1) *Lou shouted.*
Lou shouted at the kids.
Lou shouted at the kids angrily.

Rather than create three separate variations of the predicate *shouted* (each with different arity), an event variable is created which links the arguments. Thus (1) can be represented as in (2):

(2) $\exists e$ shouted(e, L)
$\exists e$ shouted(e, L) $\wedge$ at(e, K)
$\exists e$ shouted(e, L) $\wedge$ at(e, K) $\wedge$ angrily(e)

There are several reasons why this approach makes sense. The most important being that constructs in natural language seem to suggest a reification of events. Consider the pronoun *it* in (3), which refers to the event of *building a house*.

(3) *We built a house yesterday. It was hard work.*

Another reason why Davidson's event semantics makes sense is that it facilitates temporal logic, e.g. as described in Allen (1984). The reified event variable can be used to encode complex temporal constraints on the event.

The intersective treatment of event arguments is also applicable to nominal modification. Modifiers such as adjective phrases and prepositional phrases can be expressed as conjunctions on a term. An example is given in (4).

(4) *the little fierce brown mouse*
little(**M**) $\wedge$ fierce(**M**) $\wedge$ brown(**M**)

FOPC is important because it forms the basis for many other meaning representations. One popular form of FOPC used in computational linguistics today is neo-Davidsonian predicate calculus, which includes among other extensions, Parson's addition of thematic roles to the Davidsonian representation of events. I will refer to this semantic representation as neo-Davidsonian predicate calculus (NDPC) throughout the rest of the thesis. It will serve as representative of FOPC based representations for purposes of comparison.

2.2.5 Quasi-Logical Form

The Core Language Engine (CLE) is a framework for semantic interpretation of natural language sentences, as presented in Alshawi (1992). The ultimate result of natural language processing in the CLE is a comprehensive logical form (LF) that is essentially a superset of PC. The LF representation is a fully specified semantic representation of the possible meanings of a sentence.

More relevant to this discussion, however, is an intermediate representation used in the CLE called Quasi-Logical Form (QLF). The QLF is one in a number of stages in the CLE, and it is meant to be an underspecified semantic representation of the surface text. Certain language tasks have been found to be tractable using only the QLF form, versus the more fully specified LF (e.g. machine translation).

The QLF shares many properties with the model proposed in this thesis: it is derived from purely linguistic processing, it has undetermined scopal operators, and it has unresolved anaphoric references. It also includes grammatical information recovered from the syntactic analysis, such as gender and number.

There are, however, some differences between QLF and the model proposed herein. QLF is a unification based framework. It is derived compositionally from a unification based grammar and uses unification based rules for resolution to LF (and other operations). QLF also includes PC constructs in the representation (although they may be left underspecified). Finally, QLF treats nominals as unary predicates. These differences will be further detailed as appropriate in Chapter 3.

As an introduction to the QLF notation, which I will use in subsequent chapters, I will present the syntactic description and several examples from Alshawi and Crouch (1992a).

A QLF **term** (cf. PC) must be one of the following:

A variable: `X, Y, ...`

An index: `+i, +j`

A constant: `7, mary7`

An expression: `term(Idx, Cat, Restr, Quant, Reft)`

where:

`Idx`: Uniquely identifies the term expression

`Cat`: List of feature=value pairs, e.g. <type=pro, num=sing>

`Restr`: First order, one place predicate

`Quant`: A generalized quantifier, or a meta-variable if unresolved

`Reft`: A constant or term index, or a meta-variable if unresolved

A QLF formula must be one of the following:

A predicate: `Predicate( Arg1, Arg2, …, Argn)`

An expression: `form(Category, Restriction, Resolution)`

A scope: `Scope:Formula`

where:

`Predicate:`	A first or higher order predicate, or logical operator (*and*, *not*, etc.)
`Arg:`	A term, a formula, or a lambda abstract (defined below)
`Restriction:`	A higher-order predicate
`Resolution:`	A formula, or a meta-variable if the form is unresolved
`Scope:`	A meta-variable if the scope is underspecified, or a list of term indices e.g. `[ +i, +j ]` where `+i,+j` are indices occurring within `Formula`, and `+i` outscopes `+j`.

The following examples from Alshawi and Crouch (1992a) will help make the syntax more clear:

Every boy met a tall girl

```
_s:meet( term( +b, <type=q, lex=every>, boy, _q, _x),
         term( +g, <type=q, lex=a>, Y^and(girl(Y), tall(Y)), _r, _y ))
```

A resolved form corresponding to the reading *every boy met a different tall girl* can be obtained by instantiating the meta-variable quantifiers `_q` and `_r` with `forall` and `exists`, respectively, and setting the scoping meta-variable `_s` to `[+b, +g]`:

```
[+b, +g]:meet( term( +b, <type=q, lex=every>, boy, forall, +b),
               term( +g, <type=q, lex=a>, Y^and(girl(Y), tall(Y)),
               exists, +g ))
```

As another example, consider the unresolved anaphoric sentence:

Every boy claimed he met her

```
_s1:claim( term( +b, <type=q, lex=every>,boy, _q1, _x ),
           _s2:meet( term(+h1,<type=pro, lex=he>, male, _q2,_y ),
           term( +h2, <type=pro,lex=her>,female,_q3,_z )))
```

A resolved form, which assumes the term *Mary* is salient:

```
[+b]:claim( term( +b,<type=q,lex=every>,boy,forall,+b ),
[+h1]:meet( term(+h1,<type=pro,lex=he>,male,exists,+b ),
            term(+h2,<type=pro,lex=her>,
                 female,exists,mary ))).
```

From these examples it is clear that QLF, although it is closer to the surface syntactic form than the resolved LF, still is intrinsically designed to support frameworks derived from predicate calculus. The intrinsic semantic form means that testing for representational equivalency is complicated by nested lambda applications which can obscure reductions which are equivalent.

Like LFG, the CLE uses the same basic approach of attribute-value pairs used as feature constraints on syntactic and semantic categories to allow for constructive unification. Unlike LFG however, QLF does not retain grammatical function information. Instead, constituents become arguments to semantic predicates, or semantic predicates themselves. For example, the lexical modifier *most* becomes the function `ratio(x,y)`. This can have an impact on distributional and lexical processing since category information is not always retained in the representation.

Other differences arise from the level of semantic representation employed in QLF. For example, collections are represented through the `union` functor, whereas in CAMEO collections are explicitly represented as objects and can be referenced and manipulated just like singular objects. The goal of QLF is to serve as an intermediate step to a deep semantic representation, which is a higher level of abstraction than the CAMEO representation and accounts for many of the differences which I have described.

2.2.6 Robust Minimal Recursion Semantics

Minimal Recursion Semantics (MRS), as presented in Copestake et al. (2005), is a flat semantic representation which uses compositional elementary predicates (EP) and

includes a novel treatment of quantifiers. It was developed using Type Feature Structures (TFS) (although it can use other formalisms) and integrates easily into Constraint Based Grammars (CBG) because of its support of a unification operation.

The fundamental semantic units in MRS are the EPs, which encode the various lexical relations of the surface form. These EPs correspond with the semantic relations in some object language to which the MRS representation is being applied. EPs typically map to a single lexeme with arguments that encode their relational dependencies, facilitating compositional analysis.

MRS is considered syntactically 'flat' because there is no hierarchical structure. Instead, the EPs are related through variables or indices. MRS combines this flat syntactic structure with a novel representation of scope for constructs such as modals and quantifiers.

MRS produces a semantic form that explicitly constrains the possible scoping of quantifiers. This is accomplished by enumerating scopal possibilities of individual quantifiers, which allows combinatorial configurations to be later derived. Thus the resolution of scopal ambiguity can be deferred to later processing, or indefinitely in certain cases. On the other hand, when it is necessary to resolve the scope, the possibilities are constrained and easily recovered.

MRS is really a *strategy* for underspecification of quantifier scoping. In fact, Copestake et al. describe it as a "meta-level language" operating over some object language. This has traditionally been predicate calculus, but there is no restriction that necessitates this.

Copestake (2003) introduced a new approach to underspecified semantic representation, demonstrated using MRS, called Robust MRS (RMRS). The goal of RMRS is to have a single semantic representation support all levels of language processing, from shallow POS taggers through deep parsers. RMRS is an extreme form of flat semantics where all predicates are unary arity (since shallow processors will not have access to knowledge of arity), and arguments (and other operators) are expressed explicitly as binary relations. This representation allows for monotonic incremental processing that is modular. For instance, a shallow processor could construct the initial

representation, followed by a deep processor that focuses on a specific fragment of the representation.

RMRS has similar goals to the representation described in this thesis. Both address the need for a fundamental semantic representation that can be transformed into higher level form. Both are intended to be theory-agnostic, and both aspire to be computationally efficient. However, as I will explain in Chapter 3, RMRS is more ambitious in that it can be constructed from the most rudimentary processing, at the same time facilitating the higher level representation of MRS. This adds a level of complexity to the representation that would normally be mitigated by some level of representational structure. For instance, the role of predicate arguments is abstracted through a relational operator.

The strategy for robustness in RMRS results in a flat semantic description of the dependencies (analogous to the functional descriptions of LFG) composed of single argument predicate structures, argument linkage, and equivalence relations. Compositional hierarchies can then be built incrementally by manipulating the linkages.

Like QLF, RMRS is not intended as a strictly surface representation and does not explicitly encode all surface features (e.g. grammatical function). But apart from the fact that it is semantically oriented, RMRS shares the same basic functionality as QLF and other syntactic representations (including CAMEO), with respect to the surface encoding.

2.3 Distributional and Symbolic Integration

There has now accumulated a fairly large body of work applying distributional techniques to language processing. Some examples include word sense disambiguation, measuring semantic similarity, lexical semantic acquisition, and probabilistic ranking.

As the improvements on purely distributional techniques begin to level off, researchers are searching for effective ways to combine symbolic and distributional methodologies. The basic vector model based on collocations used in most distributional experiments is now being augmented with deeper symbolic information, in the form of deeper syntactic relationships.

But while deeper symbolic information is increasingly being integrated with distributional processing, applying distributional information to symbolic processing is receiving less attention. This may be partly due to the fact that it is not intuitively obvious how to do it. One recent proposal from Clark and Pullum (2007) suggests a tensor product model which amounts to multiplying the distributional vectors for words occupying the same constituent roles when comparing sentence similarities. Summing over all products would produce an activation vector which can be used as a similarity metric.

Lin et al. (2003) suggest that one problem with the distribution hypothesis is that there can be distributionally similar words that have different meanings. They propose essentially refining the distributional information using symbolic templates. For example, they describe a method for filtering a list of distributionally similar words using a measure based on how often the words appear near each other in certain symbolic patterns (under the assumption that these patterns are likely indicator of semantic incompatibility, e.g. *from X to Y*).

De Boni and Manandhar (2003) investigate how augmenting a semantic similarity measure with distributional information affects the performance of a QA system. They explore refining the semantic measure with a word frequency statistic, such that frequent words have less weight in the measure. They report improvements on a QA experiment when distributional information is included. In this case, the distributional information is used to refine the symbolic semantic measure.

Pado and Lapata (2003) extend traditional word co-occurrence vector based distributional models to incorporate syntactic dependency information in distributional lexical semantic acquisition. They introduce a parameterized generalization of dependency-based distributional vectors and show a statistically significant measure of distinguishing semantically related words.

A difficulty with this approach will always be computational complexity. Parsing large corpora for deep syntactic relations is expensive. Pado and Lapata use a scaled parser, but in principle this approach will always be more complex compared to using more shallow syntactic information.

3

A Text Representation Language

In this chapter I will develop a text representation language called CAMEO, which will be used as an experimental framework for investigating and evaluating representational properties. The CAMEO representation language is similar in some respects to other surface representations (as noted). However, CAMEO includes several key innovations which distinguish it.

The primary difference, and the initial motivation for developing CAMEO, is the extension of a sentence level representation to include the notion of recursive context, which captures structural information at the document and phrasal level. This allows text processing tasks that are normally processed at the local level to incorporate some notion of global context. Later chapters will explore possible applications of this feature.

CAMEO is a semi-flat, intermediate representation, and attempts to balance the robustness of a flat representation with the rich structure of a hierarchical dependency graph. This approach allows the grammatical functions and syntactic relations to be normalised, encoding equivalent variations in a canonical form for processing and giving the representation greater utility. For example, tasks that require specific syntactic forms can use the canonical form of the representation to distinguish them without the need for complex pattern searches, while tasks that do not depend on a specific syntactic form will be insulated from superfluous syntactic variations in the surface form.

Another unique approach of the CAMEO representation is object-orientation. Although CAMEO is a syntactic representation, it is semantically motivated and is

based on an explicit representation of events and objects. Whereas many syntactic (and semantic) representations use noun and/or noun phrases as a basis of representation for grammatical or semantic entries, CAMEO instantiates explicit objects which are then assigned properties based on the text analysis. These objects are semantic placeholders, as they represent an abstraction of a particular real-world entity. This distinction will be made clear later in the chapter, but this allows CAMEO to also be used as a rudimentary meaning representation.

The object-orientation of CAMEO includes events, which have explicit representations also and can be used as objects in certain constructions, e.g. referring expressions. (A detailed treatment of events is given in Section 3.3.2.) This explicit representation of objects and events facilitates processing for certain applications. For example, coreference resolution typically requires evaluating objects, their grammatical function, and their properties within some window of context. In CAMEO this information is expclitly encoded and organized around an object-centric approach. Both saliency table-based algorithms (Lappin and Leass, 1994), and tree-walking algorithms (Hobbs, 1977) have been implemented directly using the representation (see Chapter 4). More semantically oriented representations such as MRS and QLF cannot directly implement these algorithms because the grammatic functions are not directly retained.

Another feature of CAMEO is the integrated class lexicon which is used for deriving distributional information over word classes. Lexemes are not used directly in the representation of sentence text, as with many other representations. Instead the representation employs global lexical identifiers which reference an object in the class lexicon. This added level of abstraction simplifies distributional and other class-based processing.

Finally, CAMEO includes several other innovations which allow it to approximate a shallow semantic representation for linguistic tasks that do not need the complexity of a comprehensive semantic model. As explained in the sections below, these include a treatment of possessives, groups, and passives.

In the introduction I proposed desiderata for a text representation that, besides having a strategy for representing structural context, would be of utility as a generalized intermediate representation. Before presenting the details of CAMEO, I will first

summarize its properties relative to these desiderata. I will then present an overview of the fundamentals of the representation, followed by details of certain notable constructions, and a formal description of the syntax. I will then describe a practical implementation of the CAMEO representation with respect to a specific syntactic analyzer, and explain the process of transforming the surface text into the representation.

3.1 Properties of CAMEO

In this section I will look at each desideratum proposed in the introduction and explain how CAMEO satisfies it, in comparison to other existing representations.

3.1.1 General

CAMEO is minimally theoretic and does not constrain the analysis so deeper semantic and other syntactic representations can be derived using a direct transformation. Only the basic structure of the event and object types is imposed, and these are flexible enough to admit even minimally structured analyses, making CAMEO a general representation of broad utility.

Other representations implicitly encode theoretical bias. For example, LFG imposes constraints on its f-structures according to its syntactic theory, which would not admit ungrammatical analyses (and thus certain incremental constructions). Another example is QLF, which interprets certain semantic constructions (e.g. quantifiers) making it more difficult to transform directly into other semantic representations.

3.1.2 Rich

Since the CAMEO representation is used both to encode existing observed surface text forms, and realize novel representations, it attempts to have the widest coverage of surface forms possible, including support for some ungrammatical surface constructions. (However, these mainly occur at the higher-levels of representation, such as phrase and sentences, which are arguably easier to verify independently, e.g. it is relatively easy to verify that each sentence represented contains a main verb phrase).

The CAMEO language defines a number of atomic elements and attributes which correspond closely to the surface syntactic categories, giving it a degree of expressiveness very similar to the surface form. In addition, CAMEO defines certain attributes which control the surface realization (e.g. pre- or post- modification), which further widens its expressive coverage.

CAMEO also includes an explicit encoding of all surface features extracted from the analysis, including a recursive context type for systematic representation of extra-sentential linguistic structure. Most syntactic and semantic representations do not include contextual information at this level, and some representations do not attempt a comprehensive encoding of surface features. For example, LFG does not explicitly encode word order.

Also, because CAMEO explicitly represents all types as objects, distributional information can be attached to and derived from any linguistic head or relation, including contextual structures. This is more cumbersome in representations that don't have an integrated strategy for distributional information.

3.1.3 Reversible

Not all representations are designed to recover the original surface text through a deterministic transformation. However, this is a desirable feature for an intermediate representation because it allows some tasks to be accomplished using the representation directly, i.e. the internal result of the text processing can be converted into surface output. CAMEO is deterministic and unambiguous, resulting in a direct systematic surface realization transformation. This transformation is recursive and functions at any level in the representation, allowing surface realization of fragments and constituents at any level.

Additionally, since the CAMEO representation language does not interpret certain semantically ambiguous constructs (e.g. generalized quantifiers), it supports a canonical surface form. Other ambiguous constructs, such as prepositional phrase attachment, do not have an underspecified form in the model but are implicitly canonical. For example, the ambiguous prepositional attachment in [*I saw the man with a telescope*] has two different representations that yield the same (canonical) surface realization

In comparison, deeper semantic representations (e.g. LFG, QLF, RMRS) may require a grammar to support surface realization.

3.1.4 Incremental/Robust

Incremental processing is an approach often used in conjunction with robust methods of linguistic analysis, which are becoming increasingly important in language processing systems. A critical feature of the CAMEO representation which allows it to support robust methods is the lack of relational constraints on the principle elements. That is, there is no requirement that **obj** elements be connected to **evt** elements, and vice versa. Elements can exist arbitrarily within a context. Thus a partial parser, or other robust methods, can still be used to represent partial linguistic analysis.

Because dependencies (but not arguments) are abstracted in CAMEO, multiple levels of analysis can be combined independently into full representations. For example, the output of a noun-phrase chunker could be used first to transform all nouns into **obj** elements. This could be followed by a parser which leverages the existing representation of the **obj** elements to produce grammatical relations.

This incremental approach can also be extended to use a parallel processing paradigm where dynamic changes to the representation signal individual processing modules, which examine the representational changes and process them accordingly.

Robust incremental analysis may be problematic for other semantic representations if argument structure is required during composition (e.g. QLF). Syntactic representations may use robust processing, but integrating incremental representations from external processors may not be supported.

3.1.5 Precise and Unambiguous

Like GRs, CAMEO relations are represented directly so comparing instances of the representation is less ambiguous. Also, the original surface form is encoded, even when certain syntactic structures are normalized (e.g. passive mood). Some representations use a level of abstraction which allows multiple representations for variants of a

syntactic construction, obscuring the relationship and making comparisons more computationally complex.

CAMEO also uses globally unique identifiers on all representational objects, making the representation unambiguous at the document level, as well as the local level.

3.1.6 Flexible

CAMEO is designed to take advantage of the document object model (DOM, 2004) using an object-oriented design. All types in the representation are treated as objects with globally unique identifiers. This allows referencing any entity (not just nouns/objects) in a consistent way for linguistic processing. Manipulation is accomplished by moving objects or changing attributes, which are native operations in the DOM. For example, passifying verb phrases would only require adding the `PASSIVE` attribute on event objects. Removing appositives can be done by moving the context object containing the appositive. (Section 4.2 gives examples of manipulating the representation.)

With other representational approaches, it may be less effecient to dynamically manipulate certain properties. For example, in a predicate-based representation such as FOPC, removing an object (term) would involve searching for the term in the arguments of all predicates. To remove a constituent in a distributed representation such as GRs would require searching the dependency list for tuples that contain members of the constituent, as well as any child nodes governed by the constituent.

3.1.7 Semantic-like

CAMEO uses primitive semantic types (**evt**, **obj**, **ctx**, **mod**, **rel**), which allows it to serve as a rudimentary meaning representation for shallow semantic tasks. This extends to the integrated class lexicon which can function as a repository for class-based meaning representations extracted from a document. For example, after analysis of the sentence [*Bears are dangerous*], the [*dangerous*] relation can be copied to the [*bear*] class. Accumulating information in this way can prove useful for shallow approaches to semantic tasks such as QA. Purely syntactic representations are not suited as meaning

representations because they are syntactically constrained and only encode dependencies.

Shallow semantic information is sometimes supported by a syntactic representation, such as LFG, which includes minimal primitive semantic information, but this is primarily intended as patterns for interpretation by a semantic component. As such, representing meaning directly is not supported.

3.2 A Simplified Surface Representation

I will begin describing the CAMEO representation with a simple example, and give a brief overview of its general design.

(5) *The black dog chased the quick brown fox.*

The representation of (5) in the language is[1]:

obj[ID=*o1* DET=*the* **mod**[LEX=*black*] **class**[LEX=*dog*]]

obj[ID=*o2* DET=*the* **mod**[LEX=*quick*] **mod**[LEX=*brown*] **class**[LEX=*fox*]]

evt[ID=*e1* ACTION=*chase* TENSE=*past* S=*o1* O=*o2*]

The CAMEO representation closely resembles the element/attribute model of XML, which is used for its implementation. I use the terms *elements* and *containers* synonymously, and show them with lowercase bold letters and square brackets to denote their scope (e.g. **obj**[ ]). Elements have *attribute values*, and these will be shown in the text as NAME=*value*, where NAME is an attribute name and *value* is an attribute value (e.g. TENSE=*past*). Attributes are optional and do not have default values, i.e. if an attribute does not appear it is unspecified. (Note for certain processing some linguistic features must take values (e.g. the number feature on objects when realizing verb

[1] This is a slight simplification of the model for clarity. In fact the lex attributes (shown in the **class** and **mod** elements), actually use an identifier that references an entry in the lexis. The lexis is a lexical context that is part of the extensions to the framework and is described in Section 3.5.2.2. I will ignore this feature for the present discussion.

agreement), however these default values are deferred to subsequent processing modules).

In the example above, the subject and direct object are represented using **obj** elements, and the verb is represented using an **evt** element. Attributes and elements are used within an element's scope to complete the representation. In the example the two nouns contain the DET attribute which records the determiner, **mod** elements for the adjectives, and **class** elements for the common nouns. The verb phrase has the ACTION attribute for the verb, the TENSE attribute, and the S and O attributes to link to the subject and direct object. Each of these will be discussed at length in the sections that follow.

To further simplify the notation, I will sometimes omit the attribute names where they are obvious. For example, in the representation of **mod**[LEX=*black*], I will dispense with the LEX attribute name and write **mod**[*black*]. Italicized words in brackets, e.g. [*black*], represent a gloss of unprocessed text which would yield the correct representational element(s) when processed (in this case a lexeme). The representation of (5) above then becomes:

obj[ID=*o1* DET=*the* **mod**[*black*] **class**[*dog*]]

obj[ID=*o2* DET=*the* **mod**[*quick*] **mod**[*brown*] **class**[*fox*]]

evt[ID=*e1* ACTION=*chase* TENSE=*past* S=*o1* O=*o2*]

I will also sometimes omit attributes that are not pertinent to the discussion at hand. For instance, I will omit the attribute ID when discussing individual elements, where no reference to that element is necessary even though all elements have the ID attribute (except for **mod**s and **rel**s). The ID attribute value can be referenced from other elements as shown in (5) where the **evt** element references the objects *o1* and *o2*. (In the remainder of the thesis when discussing a specific element I will use **id**(*x*), where *x* appears as the value of the element's ID attribute.)

Elements may include other elements as shown in (5). I use the term *property* when referring to an element that occurs within the scope of another element. Thus in (8), the class [*fox*], is a property of **id**(*o2*), and the adjective [*black*] is a property of **id**(*o1*).

The primary elements (**obj** and **evt**) can also function as properties but do not appear directly embedded in other elements. Instead they must be referenced indirectly through their identifiers. This restricts the primary elements to the first level of representation in a given context, which reduces the number of representational levels required for processing.

The order of attributes appearing within the scope of an element is not significant because attributes only encode closed class lexemes or fixed syntactic properties. However, the order of elements appearing within the scope of another element is significant (in most cases). This order is directly related to the surface expression and is used in analysis and realisation. I will say more about this in section 3.2.1 below.

The attributes defined by the representation are intended to capture the surface features of the text for later use in analysis and realisation (cf. *category feature values* in Alshawi, 1992). The distinction between a lexico-syntactic component implemented as an attribute vs. an element is functional. Attributes are used to represent static features or closed class lexemes which are not given a recursive treatment in the representation. Some examples are *determiners* and *plurality* for nouns, and *tense* and *modals* for verbs.

By contrast, elements are containers and thus used to represent components that are recursive. For example, the **mod** element is used for adjectives and adverbs. The construction [*extremely loud*] can then be represented using a **mod** within a **mod** as:

mod[*loud* **mod**[*extremely*]]

The structure of elements within CAMEO was chosen to provide a trade-off between flat and highly structured representations. The principal constituents of a phrase all occupy the first level, including all objects and events. Modifiers are contained within the element they modify and so are accessible when required without complicating the basic structure. With this approach, it is clear to see the basic components of a syntactic analysis, simplifying processing and interpretation. For example, a clausal prepositional phrase would appear at the first level, making it easy to distinguish from adnominal or adverbial prepositional phrases (which would appear inside their respective elemental containers).

3.2.1 The FORM Attribute

The FORM attribute is a special attribute which can appear on any element. It is used to encode or direct the positioning of a constituent in the surface form relative to an element's parent container. Explicitly encoding the position in this manner allows a deterministic representation of surface variation like that afforded by a constituent phrase structure, while maintaining a semi-flat representational structure. Thus the positional information is available for analysis and realisation (unlike most labelled dependency-based representations), but exists as an explicit property and not an integral part of the representational structure. This is advantageous for processing tasks that need to consider the position of a constituent in the surface form because the positional information can easily be extracted, while elements can still be accessed in a position-independent manner. Additionally, this allows a simple means for manipulating surface variation during realisation, i.e. by simply changing the values of the FORM attribute on the various constituents, a wide range of surface forms can be realised.

In general the FORM attribute takes the values of *pre* and *post*, which places the element before or after its parent element. (See Section 3.3.3 for additional usages). For example, the adjective [*weary*] in the noun phrase [*the weary traveller*], is in the *pre* position, relative to the head [*traveller*]. Alternatively, the adjective phrase [*weary from the trip*] is found in the *post* position in the phrase [*the traveller weary from the trip*]. When there are multiple elements within a parent element, the elements having the same FORM value are processed in order. For example, the CAMEO representation of the sentence [*She easily does the work of three men at the company*] is:

obj[ID=*o1* PRON=*she*]
obj[ID=*o2* DET=*the* **class**[*work*]]
obj[ID=*o3* QUANT=*three* **class**[*man*]]
obj[ID=*o4* DET=*the* **class**[*company*]]
evt[S=*o1* ACTION=*do* TENSE=*present*
    mod[*easily*]
    rel[PREP=*of* OBJ=*o3* FORM=*post*]
    rel[PREP=*at* OBJ=*o4* FORM=*post*]]

Since both the **rel** elements in this example have the *post* value, the order of the elements defines which is expressed first.

The variability in the expression of verb phrases necessitates a finer granularity on the positioning of adjuncts. Initially, the representation was not able to encode the positioning of some of the more complex verbal constructions with the simple *pre* and *post* values. Verb phrases are composite constructions and adverbial modifiers are licensed in multiple slots within the phrase. For example, the sentence [*She had been reluctantly feeding the stray cat*] requires the adverbial [*reluctantly*] to be positioned within the verb phrase. Using only *pre* and *post* values for the FORM attribute would limit the surface expression to [*She reluctantly had been feeding the stray cat*] (*pre* position) and [*She had been feeding the stray cat reluctantly*] (*post* position).

To accommodate the range of adverbial positions in verb phrases, the FORM attribute was extended to include a wider range of values within the context of **evt** elements. These values correspond to the possible slots in a verb phrase and vary with the form of the verb. Some of the possible values are illustrated below.

Simple

Jim	*eats*	*fish*
pre	*post*	*postpatient*

Past Perfect

Jim	*has*	*eaten*	*fish*
pre	*postaux*	*post*	*postpatient*

Future Progressive

Jim	*will*	*be*	*eating*	*fish*
pre	*preaux*	*postaux*	*post*	*postpatient*

Modal Past Perfect Progressive

Jim	*may*	*have*	*been*	*eating*	*fish*
pre	*premodal*	*preaux*	*postaux*	*post*	*postpatient*

Ditransitive Dative

She	*gave*	*the teenager*	*the keys*
pre	*post*	*postio*	*postpatient*

3.2.2 Object Unification

An important operation for certain processing tasks such as coreference resolution is unification. Unification is the process of merging multiple elements so they can be treated as one. This includes some means for determining which elements can be unified and which elements cannot.

Unification of objects is supported in CAMEO through the implicit constraints of attributes and properties. A set of objects are compatible if they do not violate the following restrictions:

1. The plural attribute `PL` is either unspecified, or matches
2. The quantifier `QUANT` attribute is either unspecified or compatible
3. The `EXT` attribute is either unspecified or compatible
4. The gender attribute `G` is either unspecified or matches
5. The animacy attribute `A` is either unspecified or compatible

These restrictions are guidelines for unification and may be augmented depending on the processing task.

The unified objects are encoded inside a separate context using an equivalence class (see Section 3.5.2.4) which holds all references to the unified object. The representation does not take a strict interpretation of constraints on equivalence classes in order that it may remain theory-neutral. Therefore, it is possible for modules to include references to objects that do not unify in an equivalence class. This gives a corresponding representation to surface forms that violate certain restrictions, and a strategy to compensate for interpretive errors.

3.3 Fundamentals of the Representation

Elements (i.e. containers) in the CAMEO representation comprise the fundamental types of the textual representation. There are four major types used to construct the representation: objects (**obj**), events (**evt**), modifiers (**mod**), and relations (**rel**). The

primary components are the **obj** and **evt** elements. These only appear at the first level of a clausal representation and therefore do not embed in other elements directly. The **obj** and **evt** elements also carry globally unique identifiers via the `ID` attribute. The secondary elements **mod** and **rel** do not carry the `ID` attribute because they embed in a primary element and can always be identified through their parent. (Section 4.2.1 gives an example of locating a **mod** element for the purpose of removal.)

The following sub-sections describe the fundamentals of each of the major types. Section 3.4 gives deatails of how the types are used in the representation of specific syntactic constructions.

3.3.1 Objects

The basic semantic concrete, roughly corresponding to a noun phrase, is represented using the **obj** element. An **obj** is simply a container for attributes and properties. It acts as a conceptual placeholder and may be empty, thus the simplest instance of an object contains only an `ID` attribute.

When a lexical construction functions as a noun, an object is instantiated to represent it. The object is indeterminate at this point with respect to any specific semantic model, but it has a definite representation. Even if the surface noun is indefinitely determined, it is represented by a specific element not through a variable, as in predicate calculus, because there is limited semantic interpretation performed by the CAMEO representation to distinguish between these cases. Rather, the instantiated object represents the referent directly for the local syntactic context. For example, a typical representation of common nouns in FOPC based systems is shown in (6).

(6) *a book*

$\exists x\ \mathrm{book}(x)$

$\mathrm{some}(x, \mathrm{book}(x))$

This form represents [*book*] as a predicate that may be applied to any individuals in the current set or situation. The idea is that an inferencing engine could use the logical form to filter *books* from the set, and determine some specific *book* within a context. The contextual individual (i.e. *book*) may or may not have been previously instantiated

within the model. In CAMEO, the representation of [*a book*] always results in an individual (i.e. object) being created that has the class (or property) *book*, as in (7).

(7) **obj**[**class**[*book*]]

Thus, the CAMEO representation uses a placeholder for the specific *book* that the semantic interpretation might produce. This may end up being a generic object as in (8), but the representation treats these cases the same. The interpretation is deferred to later processing.

(8) *A book is a glimpse into an author's mind.*

The object element is a container for the information produced during the syntactic processing of a single noun phrase. In certain cases there may be no information available about the object, and the container is empty (e.g. the three entities referenced in the sentence [*All three arrived late*]). Otherwise, the syntactic information connected with the object is extracted and attached to the instantiation in the form of attributes and properties.

3.3.1.1 Properties

Object elements may contain various other elements as properties. The term "property" is used in this case in the syntactic sense to denote syntactic relations, and the semantic implications of these properties are left unspecified. In this section I will discuss the various properties an object element may contain.

Class

The class element is used to represent a nominal property of an object. If there are any nominally classed lexemes (i.e. common nouns) syntactically connected to an object, they are ascribed as inherited classes using class elements. For example, the simple noun phrase *tree* would become **obj**[**class**[*tree*]].

The representation does not commit to an interpretation of compound nouns. The class lexemes are simply listed in order as properties of the object. For instance [*family man*] will appear as **obj**[**class**[*family*] **class**[*man*]]. If the source analysis instead

treats the nominal compound as a single classed lexeme, it will be treated as an object inheriting from a single class: **obj**[**class**[*family man*]]. (Note the **class** element is actually implemented as a reference to an entry in the **classes** context of the language processing system. I will discuss this further in the next chapter, but for now I will ignore this distinction.)

This treatment of compound nouns is compatible with higher-level forms as in NDPC or QLF. For example, in QLF, compound nouns are represented using underspecified relations. The example given in Alshawi (1992, p. 38) for [*a computer message*] is:

```
qterm(<t=quant, p=det, n=sing, l=a>, X,
      a_form(<t=pred,p=nn>,R,[and,
             [message,X],[R,kind(Y,[computer_thing,Y]),X]]))
```

The `a_form()` specifies a "kind" relation between [*computer*] and [*message*]. Conversion of compound nouns from the CAMEO model into QLF would consist of creating QLF `a_form()`s from the elements found in an **obj** element.

Note however, that the CAMEO representation remains true to the surface form and makes no assumption about the semantic construction of compound nouns. Since **class** elements are the only representation of nominal types, the surface form of a compound noun can be reconstructed by listing all **class** elements in an object container. These elements retain their surface order, but there is nothing in the representation denoting the head.

In contrast, the QLF representation interprets the head noun of a nominal compound to construct the semantic 'kind' relation. This requires deciding whether a compound is e.g. right-headed (*water fountain*) or un-headed (*coach-player*), or deciding the correct bracketing (*plastic water bottle*). This information cannot be derived directly from the surface form, but instead must be listed in a lexical resource (see Section 3.5.2.2).

Name

The **name** element represents proper names that are syntactically connected to an object. There may be any number of these and they are represented in the order found in

the text. Again these are treated as separate properties unless the source analysis aggregates them (e.g. a named entity processor):

obj[**name**[*John*] **name**[*J.*] **name**[*Miller*]] vs. **obj**[**name**[*John J. Miller*]].

In FOPC based representations, proper names are typically interpreted as *terms* (vs. *predicates*). For instance, [*Mary*] might be given the indexed term **mary1**. This approach can make it awkward to represent syntactic constructions such as those in (9).

(9) *the unsinkable Molly Brown*
the John I knew from school

These examples require the semantic equivalent of "the person named *John*/*Molly Brown*", rather than the normal treatment of proper names. It is not clear how these constructions are dealt with in a representation such as QLF. One possibility is to treat the proper names as normal predicates, approximating semantically the class of all persons named *John*/*Molly Brown*.

The CAMEO representation avoids this complication because the object container represents a general semantic entity and serves as the repository for arbitrary properties, including proper names. The distinction of term individuals based on proper names can therefore be deferred to higher-level processing.

Mod

The **mod** element used in the scope of an **obj** element represents an adjectival phrase. They are instantiated in the order found in the text. For example, the noun phrase [*big ugly troll*] would be

obj[**mod**[*big*] **mod**[*ugly*] **class**[*troll*]].

Like adjectival predicates in NDPC representations, the **mod** elements generally imply conjunction (although the actual semantic interpretation is left unspecified). So the above example might be transformed into a typical NDPC representation as:

$$\text{big}(x) \wedge \text{ugly}(x) \wedge \text{troll}(x)$$

The notable difference with this representation is that the predicates in NDPC appear as an unordered list, whereas in CAMEO the order of the **mod** elements that occur in a container element is significant.

(The **mod** element can appear in the context of any element and is explained in more detail in Section 3.3.3).

Rel

The **rel** element used in the scope of an **obj** element represents a post-nominal prepositional phrase. For example, the noun phrase [*the book on the table*] would be represented as

obj[DET=*the* **class**[*book*] **rel**[PREP=*on* OBJ=*o1*]] **obj**[ID=*o1* DET=*the* **class**[*table*]].

(The **rel** element can appear in the context of any element and is explained in more detail in Section 3.3.4).

Like the **mod** elements described above, the **rel** elements have an implicit conjunction, analogous to the treatment in NDPC. The CAMEO representation of the example shown above might be represented in NDPC as:

$$\text{book}(\mathbf{B}) \wedge \text{table}(\mathbf{T}) \wedge \text{on}(\mathbf{T}, \mathbf{B})$$

The **rel** elements are also ordered according to the surface form, as for **mod** elements.

Inf

The **inf** element is used to connect non-finite verb phrases acting in the role of phrasal complement, with the heads they modify. The **inf** element only contains the EVT attribute which references the **id** of a non-finite verb phrase.

For instance, the noun phrase [*a good book to read*] is modelled as:

obj[DET=*a* **mod**[*good*] **class**[*book*] **inf**[EVT=*e1*]]
evt[ID=*e1* ACTION=*read* INF].

(Note the attribute INF contained in **id**(*e1*). This attribute is used to denote non-finite verbal constructions, which are used in a variety of surface representations. The **evt** element (including infinitive constructions) is described in more detail in Section 3.3.2).

In this example, the **obj** is connected to the **evt** element **id**(*e1*) using an **inf** element (which indicates a complement construction). This strategy has the advantage of treating infinitival complements like any other property (e.g. **mod**, **rel**, etc.), which simplifies the processing.

Obj

An **obj** element may contain references to other **obj** elements. Within an object container, references to other objects are accomplished using a special form of the **obj** element which contains only an IDREF attribute having the value of the referenced object's id. For example, in (10) **id**(*o1*) is referenced in the container for **id**(*o2*).

(10) **obj**[ID=*o1*]
obj[ID=*o2* **obj**[IDREF=*o1*]]

The syntactic form modelled using this representation is that of collections. The term collection here refers to a heterogeneous group of objects. The representations of collections are constructed using a parent **obj** element containing references to the members of the collection.

For example, in sentence (11) the subject noun phrase [*Dave, Bob and Andy*] is modelled in (12). The representation consists of a single collective object **id**(*o4*) containing references to three other objects **id**(*o1*), **id**(*o2*), **id**(*o3*) which are [*Dave*], [*Bob*], and [*Andy*], respectively.

(11) *Dave, Bob, and Andy found a new trail through the mountains*

(12) **obj**[ID=*o1* **name**[*Dave*]]
obj[ID=*o2* **name**[*Bob*]]
obj[ID=*o3* **name**[*Andy*]]
obj[ID=*o4* **obj**[IDREF=*o1*] **obj**[IDREF=*o2*] **obj**[IDREF=*o3*]].

(For a discussion of the collective vs. distributive reading, see Section 3.4.2 on conjunctions. Homogenous collections will be covered later in Section 3.4.9 on plural nouns. See Section 3.4.2 for details on other conjunctive constructions.)

3.3.1.2 Attributes

Attributes in general are used to capture as much of the surface information as possible. They are derived from closed class categories that can be syntactically analyzed. By giving these lexical categories special treatment during the initial linguistic processing, they can be made available to subsequent modules. Often this type of information is helpful in operations such as reference resolution and word sense disambiguation, and since these attributes are deterministic, it is more efficient to process them once initially.

There are several attributes defined for the **obj** element. Each appears only where discovered (i.e. there is no default value).

`DET` Represents determiners, e.g. *a*, *the*, *that*, *those*. Also preceding nouns of style, e.g. *Mr.*, *Mrs.*, *Dr.*, etc.

`QUANT` Quantifiers, including cardinal numbers, e.g. *some*, *much*, *more*, *many*, *most*, *41*, etc.

`PRON` The pronoun used to reference the object, if any. For example, *I*, *you*, *she*, etc.

`PERS` Person records the personal aspect of the textual reference to the object. Its value can be 1, 2, or 3, corresponding to first, second and third person. This value is relative to the nearest context element (see Section 3.5.2.1).

`EXT` This attribute represents extension quantifiers, which are certain quantifiers that come before determiners. For example, *all*, *both*, *half*, etc. (See Section 3.4.1).

In addition, there are also three attributes defined that require a deeper analysis. These attributes are also motivated by tasks such as reference resolution and word sense disambiguation, and are used to further distinguish an object. They are not ascribed to every object, since there may be cases where they cannot be determined. They are recorded whenever certain sure-fire syntactic and lexical rules are satisfied. For example, the pronoun *her* will generate a female gender attribute [`G=f`].

G	Gender records the gender (male/female/neuter) of the object
A	Animacy records whether the object is animate, inanimate, or human
PL	This attribute is used when it can be determined that an object is treated syntactically as a plural. Collections and plural common nouns are the most obvious examples.

3.3.2 Events

Events represent the primary relation among objects. The **evt** elements represent syntactic verb phrases and have optional attributes for the subject [S], object [O], and indirect object [IO] constituents. These attributes, if present, are references to existing objects in the representation. All verb phrases are treated in this manner including pleonastic '*it*' constructions [*it is raining*] and copulars [*the gate is shut*] (see Section 3.4.5).

Like surface syntactic verb phrases, **evt** elements in the representation have a wide variety of forms. I will first describe the attributes and properties used in the **evt** elements to express these forms, and then I will illustrate several of the more interesting examples before comparing this treatment to that of other representations.

3.3.2.1 Attributes

The attributes defined for **evt** elements are listed below. These attributes are intended to represent all information about the surface form of a verb phrase. Only the ACTION attribute is required – no other constraints are enforced by the representation. The grammatical and syntactic restrictions on the verb forms are expected to be enforced by the linguistic analysis (e.g. parser) or generation component.

The four auxiliary attributes (MODAL, PERF, PROG, PASSIVE) support the sixteen possible combinations of auxiliary verbs given in Huddleston and Pullum (2002, p. 105). Together with the TENSE attribute for marking the present/preterite inflection, the NEG attribute for marking the negative, and the INF attribute for marking the infinitive form, all English tenses analyzed by Burton-Roberts (1999, pp. 126-152) can be represented. Some example sentences illustrating the use of these attributes are given in Section 3.3.2.3.

Note, however, that there is no explicit information structure contained in the representation. Syntactic variations that have equivalent meanings (i.e. truth conditions) can be represented using the attributes, etc. described in this chapter, but there is no facility in the CAMEO language for indicating equivalent informational content.

S	The subject object reference
O	The direct object reference
IO	The indirect object reference
C	Complement
ACTION	Head verb uninflected form
MODAL	Modal auxiliary
TENSE	Verb tense
PASSIVE	Verb is in passive form
PERF	Verb is in perfect form
PROG	Verb is in progressive form
PART	Verb is in participle form
NEG	Verb is in negative form
INF	Verb is non-finite

3.3.2.2 Properties

The only properties allowed in **evt** elements are **mod** and **rel** elements. All other features and verb constructions are formed using the attributes described above.

The **mod** element when used inside an **evt** functions as an adverb, and modifies the head verb. For example, in (13) the adverb [*hardly*] is contained inside the **evt** element. (The **mod** element is described in more detail in Section 3.3.3).

(13) *I hardly knew her*

obj[ID=*o1* PRON=*I*]
obj[ID=*o2* PRON=*her*]
evt[S=*o1* ACTION=*know* TENSE=*past* O=*o2* **mod**[*hardly*]]

The **rel** element, when used inside an **evt** element, represents a verbal prepositional phrase. The **rel** element is comprised of the PREP attribute, which specifies the lexeme of the preposition, and an attribute specifying the object of the preposition. This latter attribute is normally an object reference, but can also be another element acting in an object capacity. For example, some parsers may represent adverbial clauses like [*after the rain stopped*] using a prepositional sense of [*after*] with the phrasal complement [*the rain stopped*]. This interpretation is dependent upon the implementation of the grammar but is supported by the CAMEO representation.

Like the **mod** element, the **rel** element also supports the FORM attribute, which allows flexibility in the syntactic location of the prepositional phrase with respect to the verb phrase. The values for the FORM attribute in relation to the **evt** element are detailed in Section 3.2.1 and are designed to allow flexibility in the positioning of the **rel** element. In general, the attribute value *pre* denotes a prepositional phrase occurring before the main verb, and the attribute value *post* denotes a prepositional phrase occurring after the main verb. The default position (when no FORM attribute is used) is after the the direct object. Examples (14) and (15) show verbal prepositions in the *post* and default (no FORM attribute specified) slots of the verb group.

(14) *She wrote in the sand a mantra.*

obj[ID=*o1* PRON=*She*]
obj[ID=*o2* DET=*the* **class**[*sand*]]
obj[ID=*o3* DET=*a* **class**[*mantra*]]
evt[S=*o1* ACTION=*write* TENSE=*past* O=*o3* **rel**[PREP=*in* OBJ=*o2* FORM=*post*]]

(15) *I ate strawberries with a fork.*

obj[ID=*o1* PRON=*I*]
obj[ID=*o2* DET=*a* **class**[*fork*]]
obj[ID=*o3* PL **class**[*strawberry*]]
evt[S=*o1* ACTION=*eat* TENSE=*past* O=*o3*
rel[PREP=*with* OBJ=*o2*]]

3.3.2.3 Examples

In this section I will present several examples to illustrate the wide range of forms of verb phrases the model supports. To clarify the notation, I will gloss the objects referenced in the **evt** elements using **obj**[*x*] where *x* is the lexical expression of the object, and other properties in a similar manner.

(16) *Caged parrots sometimes won't talk.*

evt[S=**obj**[*caged parrots*] ACTION=*talk* MODAL=*will* NEG **mod**[*sometimes*]]

(17) *Jack and Jill will be throwing Bob a party.*

evt[S=**obj**[*Jack and Jill*] ACTION=*throw* PROG MODAL=*will* O=**obj**[*party*] IO=**obj**[*Bob*]]

(18) *Lisa was frightened silly by Mark.*

evt[S=**obj**[*Mark*] ACTION=*frighten* TENSE=*past* PASSIVE O=**obj**[*Lisa*]
mod[*silly* FORM=*post*]]

(19) *The children may have been feeding the squirrels.*

evt[S=**obj**[*the children*] ACTION=*feed* MODAL=*may* PERF PROG TENSE=*past* O=**obj**[*the squirrels*]]

(16) is an example of a modal construction which includes a negative and an adverb. The adverb is in default position so it needs no form attribute.

(17) is another example of a modal, but this time the verb is in progressive form and is ditransitive. Note the order of the attributes is not significant (unlike properties).

(18) is an example of a passive construction. The subject [*Mark*] is recovered through syntactic analysis and the sentence is represented in standard form, with only the passive attribute to indicate the original construction. Removing the passive attribute would cause the same representation to generate [*Mark frightened Lisa silly*]

(19) is a complex construction that includes a modal, a perfect and progressive aspect, and the past tense. Each of these attributes is independent and may combine to represent the various possible surface syntactic forms.

3.3.2.4 Comparison with Other Representations

As mentioned in the introduction to FOPC (see Section 2.2.4), most FOPC based representations adopt a Davidsonian approach, which reifies events to allow for variable arity. In the CAMEO representation, this is not a problem because it does not have the constraints of a logical form. CAMEO is designed to be as flexible as the surface form with respect to the parameters associated with a verbal event. For example, (20) gives representations of a verb phrase for NDPC, QLF, and CAMEO.

(20) *Sally ate lunch with Steve.*

NDPC: $\text{lunch}(\boldsymbol{y}) \wedge \text{eating}(\boldsymbol{e}) \wedge \text{eater}(\boldsymbol{e}, \textbf{sally}) \wedge \text{eaten}(\boldsymbol{e}, \boldsymbol{y}) \wedge \text{with}(\boldsymbol{e}, \textbf{steve})$

QLF:
```
quant(exists, A, [lunch, A],
      [past, quant(exists, B, [event,B],
         [and, [eat, B, Sally, A], [with, B, Steve]] ) ])
```

CAMEO: **evt**[S=**obj**[*Sally*] ACTION=*eat* TENSE=*past* O=**obj**[*lunch*]

rel[*with* **obj**[*Steve*]]]

Note the sample NDPC representations I use here and throughout the remainder of the thesis, are adapted from Jurafsky and Martin (2000, e.g. p. 527). This is something of a pseudo-representation because it glosses the verb's tense (e.g. *eating*) and thematic roles. Currently there is no consensus on how to represent thematic roles, so using these high level approximations is warranted. For the purposes of exposition in this thesis, this approximation will suffice.

From the NDPC representation, it is easy to see how other forms of the verb [*eat*] with different arities can be accommodated. For example, the sentence [*Sally ate with Steve*] can be derived by removing the eaten(***e***, ***y***) predicate (along with lunch(***y***)). Similar operations can be used to produce [*Sally ate lunch*] and [*Sally ate*].

The QLF representation does not include explicit roles for the predicate [*eat*], so deriving the intransitive form requires a corresponding intransitive version of the predicate [*eat*] (or some other mechanism). However, QLF does use a reified event variable, so prepositional variants can be accommodated. To represent [*Sally ate lunch*] for example would be equivalent to removing the inner [`and`] formula, along with its second argument , leaving only the [`eat`] predicate.

In the CAMEO representation, verbal arguments are represented using attributes, and these attributes are optional. So transforming (20) into the intransitive [*Sally ate with Steve*] is accomplished by removing the verbal object attribute [`O`]. The verbal preposition is represented using a **rel** element, which can also be removed easily to produce [*Sally ate*].

It is clear that all three representations share a similar treatment of variable event arity. However, for the CAMEO representation this is accomplished by modelling the surface form, whereas QLF and NDPC rely on the reification of events inspired by Davidsonian semantics.

3.3.2.5 Infinitives

Infinitive verb phrases often appear as complements in various surface syntactic constructions. These phrases are represented in the model using **evt** elements with an extra attribute named `INF`. All other **evt** element attributes and forms apply to infinitive elements as well.

Events do not normally function as objects. Standard **evt** elements represent a finite action, which implies a temporal property. In other words, finite events happen at some fixed time reference and this property is inherent to the event. Objects, by contrast, do not have an inherent finite temporal property. To posit an object in time requires that it

be associated with some action, usually via a verb. (If the noun encompasses an action, deterministic adjuncts can place it finitely in time, e.g. [*the meeting on Tuesday*]).

For instance, the finite verb phrase [*Mark slept outside*] describes an event that takes place in the past (relative to some context). This phrase will not fit in syntactic slots that require an object: [*[*Mark slept outside*] *is fun*]. This is why finite **evt** elements may not be referenced by attributes which take objects.

But the case is different for infinitives. An infinitive verb phrase behaves more like an object, i.e. it has no inherent temporal property. For example, [*to sleep outside*] describes the idea of the act of sleeping outside, and is therefore timeless. This infinitive verb phrase *does* fit syntactic slots that require an object: [[*To sleep outside*] *is fun*]. To model this behaviour in the CAMEO representation, infinitive **evt** elements are allowed to be referenced by attributes that normally require an object.

(21) *Isabella refused to eat.*

evt[ID=*e1* ACTION=*eat* INF]
evt[ID=*e2* S=**obj**[*Isabella*] ACTION=*refuse* TENSE=*past* O=*e1*]

In (21) the infinitive verb phrase [*to eat*] is used as the direct object of the finite verb [*refused*]. The **id**(*e1*) is marked as infinite with the INF attribute, and **id**(*e2*) is otherwise a normal construction. Compare this with (22) which uses a noun as the direct object.

(22) *Isabella refused the proposal.*

evt[ID=*e2* S=**obj**[*Isabella*] ACTION=*refuse* TENSE=*past* O=**obj**[*the proposal*]]

The notion of control can be represented in CAMEO using co-indexing of **evt** attributes. However, unless the syntactic analysis explicitly marks control, it is not encoded. For example, in (21) [*Isabella*] is the subject of **id**(*e1*) (the infinitive verb phrase [*to eat*]) but does not appear in a subject attribute as in **id**(*e2*). There are several reasons to leave the control underspecified in this manner. First, the controlling subject can easily be recovered when it coincides with a constituent of the dominating phrase simply by referencing the appropriate attribute. Second, explicitly annotating the

infinitive verb phrase with the controlling subject binds it to a specific object and complicates manipulations, whereas leaving the subject implicit allows the infinitive to be freely assigned to another element simply by referencing its `ID` attribute. Finally, determining the controlling subject requires a degree of lexical semantic knowledge that is more appropriate to subsequent processing stages. For example, selecting the dominant phrase's subject or object by default is one simple heuristic, but this will fail for certain contexts such as [*She begged the fugitive to leave / She promised the fugitive to leave*]. Leaving the controlling subject unspecified allows shallow processing to use a default where necessary without constraining deeper processors that may be capable of analyzing control more thoroughly.

The CAMEO representation of infinitives is a more direct model of surface form than that given by NDPC representations, which may require the use of a higher order lambda operator. For example, the QLF representation of the infinitive construction [*It is nice* [*to live in Paris*]] is given in Alshawi (1992, p. 24):

```
[pres,
 quant(exists, A, [state, A],
  [be, A,
    [nice1_property,
     B^quant(exists, C, [event, C],
             [and, [live1,C,B], [in_location,C,paris1]] ) ] ] ) ]
```

Here the predicate [*to live*] is realized using lambda abstraction resulting in a higher order construction.

I have said events which do not have the infinitive attribute set do not behave as objects, and may not be referenced by attributes taking objects. However, a reference to a finite event can be made using a phrasal complement construction. Here the finite event participates in an independent clause, which acts as an object. In the model a clause is referenced using a context container (contexts are presented in Section 3.5.2.1). For example, [*Jorge hoped* [*the dirigible would fly*]]. In this instance a new context would be created for [*the dirigible would fly*], and this would be the referred contextual complement of [*Jorge hoped*].

The distinction between **evt** elements that are referenced by other elements becomes significant during analysis and generation because it determines the formation of a sentence in the model. An **evt** element which is *not* referenced by another element becomes the main verb phrase in a context (e.g. a clause or sentence). Other **evt** elements which *are* referenced serve as verbal complements and do not originate a clause. (This will be discussed in Section 4.1 on Surface Realisation).

3.3.3 Mods

Syntactic modifiers produce **mod** elements in the representation. These are primarily adjectives and adverbs, but can also include style nouns like [*Mr.*] and [*Mrs.*], etc. (Determiners and quantifiers are analyzed directly and denoted using attributes on the objects they modify, so this does not apply to them). As mentioned in Section 3.3.1.1, **mod** elements existing at the same level in the representation hierarchy imply conjunction, akin to the treatment given by NDPC representations.

A **mod** element is positioned inside the container element that it modifies. This provides a general and uniform representation of modifiers regardless of what constituent is being modified. Thus **mod** elements are recursive and can modify other **mod** elements.

(23) *the oblivious pedestrian*

obj[DET=*the* **class**[*pedestrian*] **mod**[*oblivious*]]

(24) *the totally oblivious pedestrian*

obj[DET=*the* **class**[*pedestrian*] **mod**[**mod**[*totally*] *oblivious*]]

(25) *the nearest oblivious pedestrian*

obj[DET=*the* **class**[*pedestrian*] **mod**[*nearest*] **mod**[*oblivious*]]

Sentence (23) shows an example of a simple modifier contained in an **obj** element. In (24) the adverb [*totally*] is represented using a **mod** element inside the adjective

element [*oblivious*]. The **mod** element is interpreted in the most local scope, so in this case [*totally*] does not modify the **obj** element [*pedestrian*]. Contrast this with (25) where both **mod** elements (ultimately) modify the **obj** element [*pedestrian*]. In cases such as this where a sequence of adjectives produce **mod** elements, the order of the elements is significant and corresponds to the surface expression of the text.

A **mod** element may only contain other **mod** and **rel** elements (**rel** elements are described in the next section). The only attributes supported by the **mod** element are the lexical id and the FORM attribute. The lexical id is a reference in the lexis context to the surface lexeme. The FORM attribute is used to record the position of the modifier in the surface text, relative to the container the **mod** element appears in (see Section 3.2.1). If the container is an **obj** element, the FORM attribute positions the **mod** (i.e. adjective) before or after the head noun. For a **rel** element, the FORM attribute determines the position of the **mod** (i.e. adverb) in the verb phrase. The FORM attribute defaults to the value *pre*, which places it before e.g. the verb. In (26), this attribute is set to *post*, which places it after the main verb. A similar approach is used to represent post-nominal adjectives (e.g. *the person responsible*).

(26) *He spoke softly*

obj[ID=*o1* PRON=*he*]
evt[S=*o1* ACTION=*spoke* TENSE=*past* **mod**[*softly* FORM=*post*]]

A **mod** element can be contained in any other type of element. When **mod** elements are attached directly to context elements, they represent adverbs acting as phrasal modifiers. For example, in the sentence (27), the adverb [*actually*] modifies the phrase [*I enjoyed kindergarten*]. A context containing this phrase would also contain a **mod** element representing [*actually*]. This is illustrated in (27) below.

(27) *Actually, I enjoyed kindergarten.*

ctx[**mod**[*actually*]
evt[S=**obj**[*I*] ACTION=*enjoy* TENSE=*past* O=**obj**[*kindergarten*]]].

3.3.4 Rels

Prepositional relations are encoded in the representation using **rel** elements. Like **mod** elements, **rel** elements are recursive, imply conjunction, and can be contained by any other element. They record the prepositional information of the syntactic prepositional phrase, positioning the object they modify in time or space.

A **rel** element usually has two attributes. The PREP attribute, which is an index to the prepositional lexeme, and the OBJ attribute, which is a reference to the object of the preposition. Sentence (28) shows an example of a **rel** element contained within an object element. The prepositional object may instead be a phrasal context functioning as a complement, as in (29). In that case the OBJ attribute is replaced by the COMP attribute.

(28) *the man under the stairs*

obj[DET=*the* **class**[*man*] **rel**[PREP=*under* OBJ=**obj**[*the stairs*]]

(29) *Roger left before the police arrived.*

ctx[ID=*t1* **evt**[S=**obj**[*the police*] ACTION=*arrive* TENSE=*past*]]
evt[S=**obj**[*Roger*] ACTION=*leave* TENSE=*past* **rel**[PREP=*before* COMP=*t1*]

In some constructions, only the PREP attribute is used, such as prepositional phrases with an adjective complement, e.g. [*at first*], or prepositional chains, e.g. [*up to*]. In these cases the only attribute is the PREP attribute, and the complement is a **mod** or recursive **rel** element contained within the main prepositional **rel** element. Sentence (30) is an example of a prepositional chain.

(30) *The slug crawled right up to the door.*

evt[S=**obj**[*the slug*] ACTION=*crawl* TENSE=*present* **mod**[*right*]
rel[PREP=*up* **mod**[*right*] **rel**[PREP=*to* **obj**[*the door*]]

Like the other elements mentioned already, when multiple **rel** elements occur within the same context, the order reflects the surface text expression. Sentence (31) gives an example of multiple prepositions modifying the same verb.

(31) *We met on a bus in the rain*

evt[S=**obj**[*we*] ACTION=*meet* TENSE=*past*
rel[PREP=*on* OBJ=**obj**[*a bus*]]
rel[PREP=*in* OBJ=**obj**[*the rain*]]

3.4 Details of the Representation

In this section I give a treatment of notable lexical syntactic classes and constructions, and give details on how they are transformed into the CAMEO representation.

3.4.1 Determiners

Determiners and quantifiers (except possessive pronouns and genitives which are treated later) are recorded as attributes on their nominal complements. These attributes have no significance with respect to the representation, since there is no difference in the representation and treatment of a definite and indefinite object. The idea of definiteness only becomes an issue for deeper processing modules such as reference resolution.

As an example, consider the indefinite reference, [*A man walks into a bar …*] In the mind of the listener a conceptual object representing the man has been created. It is indefinite in the sense that it does not represent any specific man in the listener's experience. However, the result is the same as if it was – a conceptual representation has still been created. If the sentence is followed later in the discourse by [*The man orders a drink*], another conceptual representation of a man is created. If a further stage of processing is to resolve these two objects, the determiners will play an important role and thus they are preserved in the representation. Sentences (32) and (33) illustrate how these two sentences would appear in CAMEO. A reference resolution module could later determine that **id**(*o1*) = **id**(*o2*).

(32) *A man walks into a bar.*

obj[ID=*o1* DET=*a* **class**[*man*]]
evt[S=*o1* ACTION=*walk* **rel**[*into a bar*]]

(33) *The man orders a drink.*

obj[ID=*o2* DET=*the* **class**[*man*]]
evt[S=*o2* ACTION=*order* **obj**[*a drink*]]

The situation is similar for certain quantifiers. Existential quantifiers are treated neutrally as standard determiners, because the same argument applies as for determiners. The scopes implied by these quantifiers do not need to be resolved until a logical analysis is attempted. A single conceptual object serves to represent all possible scopes. The lexical value of the quantifier is recorded in the determiner attribute as before. For example, the sentence in (34) has ambiguous scope for the quantifier [*every*], which is treated in the representation as a determiner.

(34) *Every student passed a test*

obj[ID=*o1* DET=*every* **class**[*student*]]
obj[ID=*o2* DET=*a* **class**[*test*]]
evt[ACTION=*pass* TENSE=*past* S=*o1* O=*o2*]

(35) *The student passed every test*

obj[ID=*o1* DET=*the* **class**[*student*]]
obj[ID=*o2* DET=*every* **class**[*test*]]
evt[ACTION=*pass* TENSE=*past* S=*o1* O=*o2*]

Notice that changing the values for the determiner attributes would not alter the representation as in example (35), which is not ambiguous. This is a similar approach to QLF, which utilizes the [qterm] construction to underspecify quantified terms. The QLF representation records lexical category information via features on the [qterm] and these can be analyzed by the resolution phase. In a similar fashion, the CAMEO

representation could be passed to a higher level processor for transformation into a higher semantic representation such as MRS, by analyzing the determiner attribute for the possible scopes of the quantifier.

In contrast with the treatment of generalized quantifiers, the representation *does* attempt to interpret quantifiers that actually specify quantity, such as cardinal numbers. These definite values are assigned to a QUANT attribute and can be interpreted as quantifying a homogenous collection as in example (36).

(36) *the two countries*

obj[DET=*the* QUANT=*two* **class**[*country*]]

This yields a much more syntactic approach compared with some logical forms, which interpret definite quantities using some formulaic representation. For example, the QLF representation of (36) in standard PC notation is:

λx **[** eq(***x***, **2**) **]** country(**y**)

where the relation λx **[** eq(***x***, **2**) **]** tests the cardinality of the set ***x*** and is true when it equals two (see Alshawi, 1992, pp. 16-18).

Pre-determiners like [*both*], [*all*], [*some*] are noted in a special attribute, as shown in (37). Interpretation of this attribute is left for post-processing modules.

(37) *all the horses*

obj[DET=*the* EXT=*all* **class**[*horse*] PL]

For the case where the pre-determiner is adverbially modified, the EXT attribute is embedded in a special **mod** element, which can include recursive modifiers, as shown in (38).

(38) *nearly all the horses*

obj[DET=*the* **class**[*horse*] PL **mod**[EXT=*all* **mod**[*nearly*]]]

Post-determiners such as [*many*], [*few*], and [*second*], are also treated as special **mod** elements, having a determiner attribute. This helps to distinguish them from the true determiner. Sentence (39) shows an example for the post-determiner [*second*].

(39) *The second star on the right*

obj[DET=*the* **mod**[DET=*second*] **class**[*star*] **rel**[*on the right*]].

3.4.2 Conjunctions

Conjunctions are represented explicitly, by expanding the constituents into the appropriate number of elements. For instance, a conjunction of adjectives or adjective phrases produces multiple **mod** elements and a conjunction of prepositional phrases produces multiple **rel** elements. The first element serves as a container for the others and is used to mark the conjunction. (The CONJ attribute defaults to the value [*and*] if not specified). Sentence (40) shows an example of conjoined adjectives, and (41) contrasts this with the more conventional intersective construction.

(40) *hot and tired but hungry worker*

obj[**mod**[*hot* **mod**[*tired* CONJ =*and*] **mod**[*hungry* CONJ=*but*]] **class**[*worker*]]

(41) *dangerous big green machine*

obj[**mod**[*dangerous*] **mod**[*big*] **mod**[*green*] **class**[*machine*]]

This representation is able to model arbitrary bracketing because it is hierarchical, i.e. each container delimits a bracketing context. For example, an adjective phrase is given in (42) and (43) with two bracketed syntaxes. Each can be represented depending on the hierarchical arrangement of the mod elements as shown.

(42) *red and blue, or green*

mod[**mod**[*red* **mod**[*blue* CONJ=*and*]] **mod**[*green* CONJ=*or*]]

(43) *red, and blue or green*

mod[*red* **mod**[*blue* **mod**[*green* CONJ=*or*] CONJ=*and*]]

This approach prevents having to explicitly implement conjunctive operators. It is a more computationally efficient representation because the conjunctive form is a variation on the (implicitly conjunctive) intersective form, and retains the recursive nature while allowing for arbitrary bracketing. Note that keeping the CONJ attribute with the conjoined element simplifies manipulations such as condensation (e.g. see Section 4.2), because adding or deleting conjoined elements can be done atomically, without affecting other parts of the representation.

A conjunction of nouns forms a collection (collections were briefly mentioned in Section 3.3.1.1). A collection is treated as an object containing the members of the collection. A conjunction of three nouns will yield three separate **obj** elements representing the constituents, plus a fourth **obj** element acting as a group container. Only the group element will participate in an event. For example, in (44) the subject of the **evt** element is set to **id**(*o4*), which is an **obj** element containing the other three objects. Notice **id**(*o4*) has the plural attribute (PL) set. All collections are marked plural in the representation.

The explicit representation of the group container, along with the plural attribute, can be advantageous for certain tasks such as anaphora resolution. The group container element can be processed like other objects, e.g. in a salience table, and the plural attribute will allow a coreference resolution algorithm to include any number constraints when considering the group object. Without an explicit representation of the group, resolving a referring pronoun such as *they* in this case would require an algorithm to consider possible groupings of the singular objects.

(44) *John, Paul, and George sang a song.*

obj[ID=*o4* PL
  obj[ID=*o1* **name**[*John*]]
  obj[ID=*o2* **name**[*Paul*]]
  obj[ID=*o3* **name**[*George*]]]
evt[S=*o4* ACTION=*sing* TENSE=*past* O=**obj**[*a song*]]

The ambiguity between a collective and distributive reading of (44) is retained in this representation, because the group object becomes the subject of the verb. Whether the

group [*John, Paul, and George*] sang a particular song together, or whether each member sang a different song is deferred to later stages of processing. That is, the initial representation models the form of the surface text (via the syntactic parse), which most closely resembles a collective reading. Further processing can be used to transform this into an explicit representation of the distributive reading by removing the group object, and creating a conjunction of the verb phrase by duplicating the **evt** element for each member of the group. (Representation of conjunctions of verb phrases is explained below).

A disjunction of nouns forms a disjunctive collection, which has a similar construction but uses the CONJ attribute and does not set the PL attribute. This allows a semantic processor to recover the disjunctive relationship of the members of the collection as shown in (45). Using both these forms it is possible to compose arbitrarily bracketed collections to support constructions such as [*John and Paul, or George sang a song*].

(45) *John, Paul, or George sang a song.*

obj[ID=*o4* CONJ=*or*
  obj[ID=*o1* **name**[*John*]]
  obj[ID=*o2* **name**[*Paul*]]
  obj[ID=*o3* **name**[*George*]]]
evt[S=*o4* ACTION=*sing* TENSE=*past* O=**obj**[*a song*]]

For verb phrases, a conjunction is represented by separate **evt** elements, both of which are children of the same parent context element. The CONJ attribute is used as before to support arbitrary bracketing, with the parent context element serving as the container, and the default value of [*and*] assumed when no CONJ attribute is expressed.

Usually a clausal context contains a single independent **evt** element (i.e. an **evt** that does not appear as a dependent of any other element), which serves as the main verb. In a conjunction of verb phrases, two or more independent **evt** elements are created, each having the same subject and other attributes and properties, depending on the construction. Sentence (46) shows an example of conjoined verbs sharing the same

direct object. Sentence (47) shows an example of conjoined verbs sharing only the subject.

(46) *The audience applauded and cheered the dancing bear.*

obj[ID=*o1 the audience*]
obj[ID=*o2 the dancing bear*]
evt[S=*o1* ACTION=*applaud* TENSE=*past* O=*o2*]
evt[S=*o1* ACTION=*cheer* TENSE=*past* O=*o2*]

(47) *The sailor raised the anchor and hoisted the sail.*

obj[ID=*o1 the sailor*]
evt[S=*o1* ACTION=*raise* TENSE=*past* O=**obj**[*the anchor*]]
evt[S=*o1* ACTION=*hoist* TENSE=*past* O=**obj**[*the sail*]]

This strategy of representing verbal conjunctions is a simple extension of the non-conjunctive case, and simplifies operations for manipulating the representation, since the **evt** elements can be operated on independently. For example, to change a conjoined verb phrase such as (47) into a non-conjoined verb phrase requires only deleting one of the **evt** elements. Because there is no explicit marking of the main verb phrase in a clause, either element can be removed and the remaining element is interpreted as the new main verb phrase. Simalarly, creating or extending a conjunction of verb phrases can be accomplished by adding new independent **evt** elements. Sentence (48) extends the conjunction in (47) by adding a third event element.

(48) *The sailor raised the anchor, hoisted the sail, and headed to sea.*

obj[ID=*o1 the sailor*]
evt[S=*o1* ACTION=*raise* TENSE=*past* O=**obj**[*the anchor*]]
evt[S=*o1* ACTION=*hoist* TENSE=*past* O=**obj**[*the sail*]]
evt[S=*o1* ACTION=*head* TENSE=*past* O=**obj**[*sea*]]

3.4.3 Dependent, Coordinated, and Relative Clauses

Clausal constructions are represented using the context (**ctx**) element. In Section 3.5.2.1 I will explain how the context element is used within the global structure of the CAMEO representation, but here I will describe some of its uses at the local sentential level.

Every sentence is represented within a **ctx** element. This extends to embedded sentences (clauses) as well. A **ctx** element is objectified like other elements, and has a globally unique identifier which can be referenced in various constructions.

A subordinate clause is represented as any other sentence, but is contained within (actually referenced by) the dominant clause. In example (49), the dependent clause [*dinner was ready*] is placed inside the dominating event, in this case [*Beth said*]. The complementising conjunction is encoded as a CONJ attribute.

(49) *Beth said that dinner was ready.*

ctx[**evt**[S=**obj**[*Beth*] ACTION=*say* TENSE=*past*

ctx[CONJ=*that* **evt**[S=**obj**[*dinner*] ACTION=*be* **mod**[*ready*]]]]]

Coordinated clauses are treated in a similar manner. Each clause produces a separate instance in the representation, and the conjoined clause appears *within* the context of the original. The conjunction is also recorded in this case as an attribute on the conjoined clause. This is done so the same processing and representation can accommodate both subordinate and conjoined phrases. An example is shown in (50).

(50) *Mark offered her money, but she wouldn't take it.*

ctx[**evt**[S=**obj**[*Mark*] ACTION=*offer* TENSE=*past* IO=**obj**[*her*] O=**obj**[*money*]]
ctx[CONJ=*but*
evt[S=**obj**[*she*] ACTION=*take* MODAL=*would* NEG O=**obj**[*it*]]]]

A relative clause is consistent with the previous approach. The relative clause is contained within the object it modifies, with no conjoining attribute, as in example (51).

(51) *The man who bought the ticket is gone.*

ctx[**evt**[S=**obj**[*the man*

ctx[**evt**[S=**obj**[*who*] ACTION=*buy* TENSE=*past* O=**obj**[*the ticket*]]]

ACTION=*be* **mod**[*gone*]]]

The CAMEO representation of clauses, like the approach taken for conjunctive **mod** elements, gives a uniform and computationally efficient representation. That is, conjunctive phrase constructions can be represented and processed in a similar manner to embedded phrases, and all recursive elements have a similar form.

3.4.4 Genitives and Possessive Pronouns

The possessive relationship is represented using the **prop** element, which references some possessed object. For example, (52) shows two (unspecified) objects in a possessive relationship, where **id**(*o2*) possesses **id**(*o1*).

(52) **obj**[ID=*o1*]
obj[ID=*o2* **prop**[OBJ=*o1*]]

Simple possessive syntactic forms such as [*Bill's hat*] fit easily with this approach, where there is a possessive relation between two objects. But because the semantics of the relationship are underspecified, this representation also holds for more abstract meanings, such as the case where [*Bill*] is the maker and not the owner of the [*hat*]. Other examples include [*summer's heat*] or [*Julie's friend*], where the possessive relationship is more ambiguous. A similar argument was made for the case of compound nouns (see Section 3.3.1.1 on *Class* above), where the relationship between compound class nouns is left unspecified. Notice that using the possessive object representation here distinguishes the surface form of [*summer's heat*] from the nominal compound [*summer heat*]. Semantically the difference may be unimportant, but that is left for higher-level processing to interpret.

The representation is constructed as follows. When a genitive or possessive pronoun modifying a noun is encountered, two **obj** elements are created. The first is an element representing the head noun as described in Section 3.2.1. The other element is used to

represent the possessive noun. Even though the reference to this noun is oblique, the possessive noun must still be an object and thus merits a representation. The two **obj** elements are linked in a possessor/possessed relationship via a property element (**prop**) placed inside the possessor object which references the possessed object. A simple example is shown in (53).

(53) *Her house*

obj[ID=*o1* **class**[*house*]]
obj[PRON=*her* **prop**[OBJ=*o1*]]

By comparison, higher-level semantic representations which utilize lexical semantic resources, will attempt to do some interpretation on the lexical terms to distinguish cases like those shown in (54) and (55)

(54) *Luke's father*

father_of(**Luke**, ***x***)\

(55) *the kitten's paw*

kitten(***y***) ∧ paw_of(***y***, ***x***)

Modifiers such as genitives are recursive and more complex constructions are thus possible. However, each genitive produces an object with a property to the next object in the syntactic chain. Example (56) shows a (somewhat contrived) example and the resulting representation.

(56) *His brother-in-law's sister's cousin's podiatrist*

obj[ID=*o1* PRON=*he* **prop** [OBJ=*o2*]]
obj[ID=*o2* **class**[*brother-in-law*] **prop** [OBJ=*o3*]]
obj[ID=*o3* **class**[*sister*] **prop** [OBJ=*o4*]]
obj[ID=*o4* **class**[*cousin*] **prop** [OBJ=*o5*]]
obj[ID=*o5* **class**[*podiatrist*]]

Genitives can also be applied to collections to produce both possessor collections and possessed collections, in the same recursive structure. Example demonstrates a conjunctive collection functioning in both a possessor and posessee relationship.

(57) *His brother and sister's dog*

obj[ID=*o1* PRON=*he* **prop** [OBJ=*o2*]]
obj[ID=*o2* CONJ=*and* **prop** [OBJ=*o3*]
obj[ID=*o1* **class**[*brother*]]
obj[ID=*o2* **class**[*sister*]]]
obj[ID=*o3* **class**[*dog*]]

This flat approach of representing the possessive relation through the **prop** element is dictated by the semantic orientation of the representation, i.e. genitives and possessives are no different than other objects and therefore must occupy the same level. The advantage of this arrangement is that it simplifies tasks such as anaphora resolution (since each object in the genitive chain is explicit and can be easily enumerated), while still marking the possessive/genitive relation. Using a hierarchical representation would negatively impact processing, since nouns could then appear at all levels in the representation, not just the first.

The property element allows a recursive approach to realising possessive constructions. During realisation, an object is checked to determine if it is possessed. If a possessor is found the possessor is realised first. Applying this recursively will realise the correct surface form of arbitrary possessive constructions from the representation.

3.4.5 Copular Constructions

Copular constructions including predicative adjectives are not interpreted and reduced to their semantic equivalent, as in some representations. Instead they are treated as events similar to other verbs. For example, in some representational approaches, sentences such as (58) and (59) are represented as predicates over a single instance variable.

(58) *Sylvia is a necromancer*

necromancer(**Sylvia**)

(59) *The river is wide*

river(**R**) ∧ wide(**R**)

(60) **evt**[S=**obj**[**name**[*Sylvia*]] ACTION=*be* O=**obj**[DET=*a* **class**[*necromancer*]]]

(61) **evt**[S=**obj**[DET=*the* **class**[*river*]] ACTION=*be* **mod**[*wide*]]

Contrast this with the representations in (60) and (61) for the CAMEO representation. The CAMEO representation is closer to the QLF, which uses states to represent copular constructions. States under QLF function much the same as events, allowing for participation in arbitrary constructions through reification. Without this reification, verbal prepositions on copular constructions become problematic. For example, the sentence [*The river is wide by the sea*] would be difficult to represent using the approach in (59) because the preposition [*by the sea*] would most likely have to be linked with [*river*], which would not capture the entire meaning. Using the **evt** element as in (61) allows the preposition to modify the event rather than the object, essentially representing the meaning as "the event of the river being wide happens by the sea". The same mechanism would allow other event semantics to apply to copular constructions, such as temporal logic.

3.4.6 Passive Construction

In contrast to copular constructions, passive verb constructions are interpreted and represented using a canonical form. An attribute (PASSIVE) is set on the **evt** element to record the passive voice so that the passive surface form can be recovered when necessary. Otherwise, there is no difference in the representation compared with other non-passive events. This strategy prevents having to support a specialized form for passive events in post-processing modules.

Decoding the passive form in the model consists of setting the subject (if it exists) and object of the verb correctly. For example, the sentences (62) and (63) both share the representation in (64) (except for the PASSIVE attribute):

(62) *A good time was had by all.*

(63) *All had a good time.*

(64) **obj**[ID=*o1* DET=*all*]
obj[ID=*o2* DET=*a* **mod**[*good*] **class**[*time*]]
evt[ACTION=*have* TENSE=*past* S=*o1* O=*o2* PASSIVE]

Giving a literal interpretation of (62) would produce the representation shown in (65). Here the sentence is represented as a past participle (via the PART **evt** attribute), with a verbal preposition.

(65) **obj**[ID=*o1* DET=*all*]
obj[ID=*o2* DET=*a* **mod**[*good*] **class**[*time*]]
evt[ACTION=*have* TENSE=*past* PART S=*o2*
rel[PREP=*by* OBJ=*o1*]]

The advantage of normalizing the passive form of the sentence, rather than representing the passive construction, is that it allows passive sentences to be processed in the same way as non-passive sentences. Without this normalization, shallow tasks may incorrectly process passive constituents. For example, distributional processing will include the subject of a passive verbal construction with non-passive verbal subjects. This could possibly degrade the distributional data because a passive subject actually receives the action of the verb, rather than initiates it as with a non-passive subject. For instance, distributionally determining objects that can *eat* might erroneously include *cake* if the passive sentence [*The cake was eaten by the children*] is found in the corpus. If the representation does not intrinsically normalize passive sentences, each processing task will need to implement its own interpretation of passive construction, or risk incorrectly analysing passive constituents.

There are several cases where it is difficult to correctly identify passive constructions, and this is one of the difficulties with a normalized representation. A general rule for adopting the verbal prepositional complement as the passive subject will incorrectly identify prepositions acting in other capacities such as locative or instrumental. For example, *the sea* will be incorrectly realized as the subject in the passive sentence [*The ceremony was held by the sea*]. Additionally, several

prepositions can be chained together making it difficult to recover the passive subject correctly as in [*The victims were rescued by helicopter by the army*].

3.4.7 Dative Constructions

Unlike the normalization of passive constructions, the alternate forms of ditransitive verbs are represented directly. The dative alternation is represented with the indirect object recovered explicitly, and the non-dative (prepositional) form is represented in the model using a standard **rel** element. Examples (66) and (67) show instances of the dative and non-dative representations (respectively) in the model.

(66) *Mark told Mary the news*

evt[S=**obj**[**name**[*Mark*]] ACTION=*tell*
IO=**obj**[**name**[*Mary*]] O=**obj**[DET=*the* **class**[*news*]]

(67) *Mark told the news to Mary*

evt[S=**obj**[**name**[*Mark*]] ACTION=*tell*]
O=**obj**[DET=*the* **class**[*news*] **rel**[PREP=*to* OBJ=**obj**[**name**[*Mary*]]

This strategy simplifies the process of transformation into the representation because, like the interpretation of the passive construction, the preposition in a non-dative ditransitive construction can be ambiguous. For example, in the sentence [*The owner took his dog to the vet.*], the preposition does not mark an indirect object. Replacing the verb *took* with *gave* changes the function of the preposition so that it does mark the indirect object. Distinguishing these cases would require lexical knowledge of verbs that license ditransitive constructions.

Although normalizing the non-dative construction would be advantageous for the same reasons as normalizing the passive construction, the non-dative construction does not have the same disadvantages when represented directly. There is no shift in grammatical function for constituents in a dative construction, as appears with the passive form. For this reason, it is less important for the normalization of dative constructions to be incorporated in the representation, and the requirement for lexical resources makes the dative normalization prohibitive.

3.4.8 Reflexives

The representation attempts to interpret reflexive constructions thereby instantiating a single **obj** element to represent both reflexive references. The reflexive pronoun is normalized and added as a PRON attribute on the object. This preserves the gender information of the reflexive pronoun for later use. Like the treatment of the passive verb alternation, encoding the reflexive construction alleviates some of the complexity for later processing stages. In this case, one less object is included in the representation, reducing the ambiguity for modules such as coreference resolution.

Although reflexive pronouns are often explicitly tagged by the morphological analyser, the syntactic analyser may not indicate the reflexive relationship (beyond normal syntactic representation). However, the majority of reflexive constructions do not require a sophisticated interpretation to recover and can be accommodated in the rules of the representational transformation. Sentence (68) shows an example of the representation of a reflexive construction.

(68) *Zachary congratulated himself*

obj[ID=*o1* **name**[*Zachary*] PRON=*he*]
evt[S=*o1* ACTION=*congratulate* TENSE=*past* O=*o1*]

3.4.9 Plural Nouns

The treatment of plural nouns has already been introduced in Section 3.3.1.1 on conjunctions (see example (11)). I explained how heterogeneous collections are treated using a single object as the group container, with the plural attribute set. Homogenous collections are also represented as object elements with the plural attribute (PL) set. However, these objects do not act as explicit containers for other objects. The plural attribute is the only indication that they are plural.

Homogenous collections may either be plural class nouns [*dogs*], [*trepanners*], or unspecified collections [*The Board of Regents*]. There is no real difference among these representations -- each represents a collection of objects. In the case of class nouns the members are implied. In the case of unspecified collections the members are unknown.

Examples (69) and (70) show representations of a plural class noun, and an unspecified collection, respectively.

(69) *the dogs*

obj[DET=*the* **class**[*dog*] PL]

(70) *The Board of Regents*

obj[**name**[*The Board of Regents*] PL]

This approach is similar to QLF, where the category features of a qterm carry the singular/plural attribute.

In the CAMEO representation, if a quantity is used to modify the collective noun, it is noted in the QUANT attribute. If present, the QUANT attribute gives the size of the collection. This may be a numeric value (*three, 101)*, or something more vague (*few, much*). An example of a plural class noun with numeric modification is shown in (71).

(71) *500 dingos*

obj[QUANT=*500* **class**[*dingo*] PL].

(Definite quantification was previously discussed in detail in Section 3.4.1 on determiners.)

3.4.10 Complements

Some syntactic constituents allow for phrasal or verbal infinitive complements. Examples include adjectives (*sad* [*to leave*]) and certain nouns (*the hope* [*Spring will arrive early*]). Verbal infinitives were already discussed in Section 3.3.2.5. As I showed, the ID attribute on an infinitive event can be referenced as a constituent in finite verbal events. When infinitives complement a non-verbal constituent, the infinitive **evt** element is included within the scope of the container element through the use of indirection, by adding a placeholder **evt** element which references the infinitive **evt** element's ID. This allows infinitive **evt** elements to remain at the same level as other **evt** elements in the context, and still treat inifitive modifying complements similar to other

modifiers such as **mod** or **rel** elements. Example (72) shows a sentence with an adjective phrase (**mod** element) with an infinitive verbal complement.

(72) *Hal is happy to comply*

evt[S=**obj**[*Hal*] ACTION=*be* **mod**[*happy* **evt**[ACTION=*comply* INF]]]

Phrases acting as complements are referenced by the ID attribute of the local context element that contains them (context elements are detailed in the next chapter). Using the COMP attribute set to its context ID, a phrase can complement a constituent in the same manner as an infinitive verb phrase. Example (73) shows the representation for a noun with a phrasal complement.

(73) *the fact the defendant was guilty*

obj[DET=*the* **class**[*fact*] **comp**[*the defendant was guilty*]]

3.5 Extensions

The CAMEO language described in the previous sections is designed for the representation and processing of text at the sentence and phrase level. To extend the representation beyond this basic level of analysis requires a treatment of other aspects of textual documents, which I will briefly introduce before describing the specific features and extensions added to CAMEO to address these properties. These extensions are designed to tightly integrate the lexical, linguistic, and pragmatic components of text processing.

3.5.1 Motivation

One important aspect of semantic processing is lexical semantics. Beyond providing for the basic lexical representation of words, to be useful for semantic processing, a text representation should have some means of organizing and integrating extended lexical information, i.e. the LKB. Using a common representation for the LKB and other

representational components has the advantage of simplifying the sharing of information among processing modules, and reuse of supporting utilities.

Additionally, the support of a lexical component should allow for the collection and integration of distributional information. For instance, information about word frequencies, collocations, and other distributional events should be easily accessible from within the representation. This facilitates the integration of distributional processing techniques concurrently with other symbolic processing.

For the representation to allow for more advanced analysis beyond the sentence level, a strategy for representing contexts is also necessary. For logical semantic processing, disambiguation is often relative to some context. An explicit representation of contexts facilitates transformation into a contextual logic (e.g. Buvač, 1996). Additionally, text has organization apart from grammar and syntax and this is useful information that should be made available to processing modules. For instance, documents are sometimes organized with chapters, sections, etc. These may have titles or other marked text which can be given more weight during analysis if context is considered, rather than processing them simply in line as free text. In Chapter 4 I will develop the contextual representation of document structure in detail, and in Chapter 5 I will explore contextual issues in symbolic processing.

3.5.2 CAMEO Extensions

The following sections extend the basic CAMEO representation to include element containers for the **lexis**, **classes**, **assert**, and **process** contexts. These elements provide a uniform representation and structure for distributional and contextual processing.

3.5.2.1 Contexts

One of the most important novel features of the CAMEO representation is the inclusion of a generalized representation for contexts. A CAMEO context is a local space that defines the syntactic, pragmatic, and semantic reference point for a fragment of the representation. Contexts were briefly mentioned previously as container elements

for sentences and phrases. In the extended representation, contexts are expanded to include all document components and organizational structures. Every logical grouping of text, whether implicit or explicit, can be marked using the context element.

There are several advantages to having this flexibility for representing contexts in the representation. Since a context is a container for other elements, it provides an anchor or reference for otherwise unconnected linguistic events. For example, in the case of sentence fragments or a failed or partial parse, there will often be orphaned NPs that are not lexically connected with other syntactic components. Some representations require all lexical components to be connected (directly or indirectly) (e.g. Trujillo, 1995, p. 90). The CAMEO representation does not have this restriction as elements are independent and can appear anywhere inside a context. The context element provides a default relation between these otherwise unconnected linguistic objects in the representation.

This lack of constraints on the relations among elements in the CAMEO representation means there are no special requirements on the initial processing. Robust and incremental methods can be used to translate free text into a CAMEO representation, including partial parsers, noun chunkers, and named entity recognizers. The use of contexts gives enough structure to the representation that processing using these types of models can be leveraged effectively.

Using a contextual model also provides reference frames for coreference resolution and other processing tasks that can take advantage of contextual information. In the next section I describe the organization of the various contextual elements that make up the representation in the system.

Contextual Hierarchy

Figure 3.5.2.1 shows a diagram of the contextual hierarchy in the representation. The highest level context is the root context, which contains all components of a CAMEO representation. No other context may contain a root context. The root context contains a single instance of the **lexical** context, the **classes** context, the **assert** context, and the **processing** context. The **lexical** context is used to implement the lexical database. It contains the individual lexemes available in the representation. The **classes** context is

used to hold lexical class information. The **processing** context is used to store dynamic information from processing the **assert** context, such as coreference resolution. The **lexical**, **classes**, and **processing** contexts will be discussed in the following sections. The **assert** context is the container for representations of textual entities. It may contain any number and type of contexts except the **root**, **lexical**, and **processing** contexts. New text representations are placed in this context for further processing.

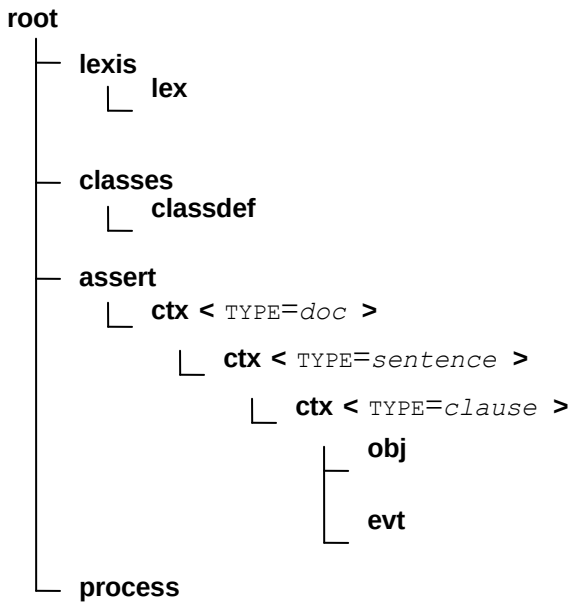


Figure 3.5.2.1 – Contextual hierarchy of CAMEO

A new context is created for each document or text entity that is to be processed in the system. The context specifies the type of text entity it was derived from (document, dialogue, book, news article, etc.). Other pragmatic information may be added depending on the sophistication of the input pre-processing and the source data. For instance, the source data may include mark-up for a reference URL, an author, the date of publication, etc. If these are not explicitly marked in the document, they might appear as part of the free text. In that case they could be processed after the initial representation and later moved back into the context header. (See Section 5.7 5.7.2for examples of representations using the contextual elements described in this section).

Further contexts may be created depending on the document type and the source text. Contexts may be created for book chapters, scenes in a play, captions, footnotes,

turns in a dialogue, etc. These types of contexts are abstract containers and do not have an inherent treatment in the representation, other than providing the facility for marking the representation for reference and further processing.

The next fundamental structure of context is the sentence. A sentential context is used to mark every formal segment of text that should be processed independently. For documents this is a full and complete sentence, or the closest approximation. For spoken corpora this could be an utterance. For other structured text it could be an element of a larger collection. The segmenting of the document into these units is a committed processing decision that must precede all others. If multiple sentence segmentation algorithms are used, separate copies of the entire document context must be included.

The sentential context is the basic unit of reference in the representation. In order for multiple independent processing sources to operate on the data, some means must exist for a common reference to the source text. Each sentential context thus includes the original source text. Each word in the sentence can then be uniquely indexed relative to the context. This also provides a means for developing and testing independent stages of a system of processing modules.

For instance, a text simplification module may split a sentence into two or more, and a named entity module may then be run on the original sentence. The original reference text can be used to correlate the named entities with the new text. Note that the text simplification module would create two new sentential contexts (with appropriate identifiers) in the original sentential context.

The basic local linguistic processing occurs within the context of a sentence. Usually this includes morphological processing, part of speech tagging, and parsing. These functions might be processed independently using the representation to store state information, or they may be integrated in a single processing component. At some point the sentential text needs to be transformed into the CAMEO representation.

The last type of context defined is the phrasal/clausal context. This context has the smallest scope and corresponds to a phrasal unit in a phrase structured grammar. It is larger than a noun or verb phrase, but not necessarily a complete sentence. Phrases are

used to represent the context of a complement, adjunct, parenthetical comment, etc. Phrasal contexts mirror the structure of natural language and are thus freely recursive, allowing a treatment of quotes within quotes, as well as prepositional chains and other recursive constructs.

If a sentence includes another complete sentence, such as quotations, dialogue or sentential complements, the phrasal context is used rather than a new sentential context. This is to distinguish between a sentential context that references the source material, and a sentential context that has been decided during processing (which must use a phrasal context). Thus by iterating over sentential contexts, a processing module can traverse the original sentences in a document.

As mentioned earlier, sentential contexts *can* be embedded in other sentential contexts, but only when a processing component has created new material (not simply transformed the existing text). In the example of text simplification, it is likely that the new sentences will include new textual content. In order to ensure it is referenced with the original text the new sentential contexts are embedded in the original sentential context.

Multiple analyses of a sentence can be included in a sentential context as sister phrasal contexts. However, there is currently no inherent support for distinguishing these, so independent processing modules expecting multiple analyses would have to treat these as a task-specific representation. Standard processing uses the first phrasal context within a sentential context as the active analysis.

3.5.2.2 Lexis Context

The **lexis** context is a container for all lexical information used in the representation. Each entry in this context is a **lex** element representing a lexeme. The **lex** element is comprised of an `ID` and a graph of the unmorphed stem of the lexeme.

The `ID` of the **lex** elements are referenced by the other elements in the representation (except for **class** elements as explained below). Using this level of indirection helps to facilitate collection of distributional information for statistical processing by providing a

globally unique numerical identifier which is used throughout the corpus in each instance of the lexeme.

3.5.2.3 Classes Context

Besides the graphs of the individual raw lexemes, the representation includes a context for lexical class information. Each entry in the **classes** context is a **classdef** element which potentially contains lexical semantic information about a nominal class. The **classdef** element has an `ID` attribute which is referenced by **class** elements residing inside an **obj** element. (The **class** element is used to represent a common noun as discussed in Section 3.3.1.1).

Figure 3.5.2.3 shows a representational fragment illustrating the relationship between a lexeme, a class definition, and an object of that class. Within the **assert** context of this example there is a single **ctx** element containing a single **obj** element. The **obj** element contains a **class** element referencing the **classdef** defined within the **classes** context. Note that there are two classes defined, each referencing the same lexeme [*bank*]. This is an example showing how multiple senses for a lexeme can be represented. The `ID` attribute of the class definition uniquely identifies the sense of a lexeme. In this example the object has been determined to be using the second sense (`ID`=*c2*) of the [*bank*] class.

```
lexis
[
    lex[  ID=123 GRAPH=bank ]
]

classes
[
    classdef[  ID=c1 LEX=123  ]
    classdef[  ID=c2 LEX=123  ]
]

assert
[
    ctx
    [
        obj[  class[  IDREF=c2  ] ]            ]
    ]
]
```

Figure 3.5.2.3 – Sample representation of class definitions

Using the **classes** context, lexical semantic information can be integrated into the framework. Each **classdef** element is a container which can potentially include lexical semantic information about a particular sense of a lexeme. For example, information on the qualia structure of a noun (Pustejovsky, 1991) could be represented using attribute value pairs within the **classdef** element. Note that other syntactic categories (besides common nouns) could be included in the **classes** context, however currently only common nouns are treated in this manner.

Statistical information for individual classes can be extracted from the representation in the same way as for raw lexemes. The class `ID` attribute can be used to search the assertional context for instances of the class, and from this distributional information can be collected. Individual processing modules can then store arbitrary statistical information inside the class container using proprietary elements. (For a more detailed discussion see Chapter 6).

3.5.2.4 Processing Context

The CAMEO language is designed to be a dynamic, incremental representation. The **processing** context of the representation is used to maintain information about the state of a process. Running various processing modules on a text entity has a cumulative

effect and the state of the system is preserved across processing instantiations. In this way the state evolves as more information is processed.

The principle element in the processing context is the **eq** element, which contains an equivalence class for an **obj** element (i.e. individual) in the representation. An equivalence class contains links to all the references made to an individual in the assertional context. This requires a coreference resolution module to process the assertional information and create the equivalence class. Note, however, that this is not limited to a single document but can run over the entire assertional context. Equivalence classes link sentences about an individual throughout the assertional context, providing a means for semantic processing beyond the local syntactic compositional level.

As coreferences are accumulated inside an equivalence class, the information about the corresponding individual evolves. Thus, the equivalence class becomes a container for the knowledge discovered about an individual in a text (or texts). Semantic processing could then be applied to interpret the information in an equivalence class and derive semantic properties of the individual. This would produce a function similar to the *profiles* described in Bergler (1995, pp. 111), which are collections "of all properties that a text asserts or implies about a particular discourse entity." The advantage of using CAMEO over *profiles*, besides the integration with other aspects of processing, is that the equivalence classes collect syntactic and pragmatic information through the object references, as well as derived semantic properties.

The processing context is also the container for general world knowledge (i.e. KB). The same representation as employed in the **assert** context can be used to represent this information, although higher level semantic derivations would probably be necessary. The reason this type of information belongs in the processing context, not the assertional context as might be expected, is because it is somewhat dynamic in nature. In fact, the intent is to allow the semantic processing of the assertional context to derive some of this general world knowledge. As such, it makes more sense to segregate this internally derived information from the more static, imported assertional information.

However, it should be noted that dynamic information is not limited to the processing context, and one of the strengths of the representation is that it facilitates a black-board between processing modules. Results from intermediate processing tasks

can be stored using elements or attributes on the representation in the assertional component, for later use by other processing tasks.

3.6 Formal Syntax of the CAMEO Language

The CAMEO representation language is implemented using XML, and can therefore be shown to be a regular context-free language (Berstel and Boasson, 2000). The CAMEO syntax is based on elements and attributes, where elements are freely recursive and attributes are not. This section gives the formal syntax for the CAMEO representation language.

3.6.1 Formal Syntax

The syntax of the CAMEO representation is given in abstracted EBNF below, following the typographical conventions used throughout the thesis. Non-terminals are printed using capitalized italics, e.g. *Relation*. Terminals are printed using lowercase (or small-capitals) font, e.g. **obj,** QUANT, and *several*. The EBNF operators used in the notation are: exclusive OR (|), zero or more (*), zero or one (?), and one or more (+).

Top ::= *Root*
Root ::= **root**[*Lexis Classes Assert Process*]
Lexis ::= **lexis**[*Lex**]
Classes ::= **classes**[*ClassDef**]
Assert ::= **assert**[*TopContext**]
Process ::= **process**[*EqClass**]
Lex ::= **lex**[ID=*Id* GRAPH=*Graph*]
ClassDef ::= **classdef**[ID=*Id* LEX=*Graph*]
TopContext ::= **context**[ID=*Id* *TopContextType* (*OrgContext* | *SentenceCtx*)*]
EqClass ::= **eq**[ID=*Id* *ObjectRef**]
OrgContext ::= **ctx**[ID=*Id* *OrgCtxType* (*OrgContext* | *SentenceCtx*)*]
SentenceCtx ::= **ctx**[ID=*Id* TYPE=*sentence* (*ClauseContext* | *Object* | *Event* | *Relation* | *Modifier*)*]
ClauseContext ::= **ctx**[ID=*Id* TYPE=*clause Conjunction*? (*ClauseContext* | *Object* | *Event* | *Relation* | *Modifier*)*]
TopContextType ::= TYPE=(*doc* | *book* | *article* | *dialogue* ...)
OrgCtxType ::= TYPE=(*chapter* | *section* | *scene* ...)
Conjunction ::= CONJ=(*and* | *but* | *or*)
Object ::= **obj**[ID=*Id* *ObjFeatures* *ObjectRef** *ClassRef** *Modifier** *Relation**]
Modifier ::= **mod**[LEX=<*lex identifier*> *ModFeatures* *Modifier** *InfinitiveRef** *Relation** *Complement**]

Relation ::= **rel**[*Preposition ObjectRef ComplemenReft*? *Form*? *Modifier** *InfinitiveRef** *Relation** *Complement**]
Event ::= **evt**[ID=*Id Action Subject Object IndirectObject ComplementRef EventFeatures Relation** *Modifier** *InfinitiveRef**]

ObjFeatures ::= *Quantifier*? *Plural*? *Determiner*? *Gender*? *Animacy*? *Pronoun*? *Question*? *Person*? *EqClassRef*?
ModFeatures ::= *Quantifier*? *Plural*? *Determiner*? *Negative*? *Form*?
EventFeatures ::= *Tense*? *Perfect*? *Participle*? *Progressive*? *Passive*? *Modal*? *Negative*? *Infinitive*? *Form*?

EqClassRef ::= EQ=<*equivalence class identifier*>

ClassRef ::= **class**[IDREF=<*class identifier*>]
ObjectRef ::= **obj**[IDREF=<*object identifier*>]
InfinitiveRef ::= **inf**[EVT=<*evt identifier*>]
ComplementRef ::= COMP=<*context identifier*>

Id ::= <*unique identifier*>
Quantifier ::= QUANT=(*many* | *few* | *more* | *several* | *1* | *2* | *3* ...)
Plural ::= PL
Determiner ::= DET=(*a* |*the* | *this* | *that* ...)
Gender ::= G=(*f* | *m* | *n*)
Animacy ::= A
Pronoun ::= PRON=(*he* | *she* | *it* |*they* ...)
Question ::= QUEST=(*who* | *what* | *where* |*why* ...)
Person ::= PERS=(*1* | *2* | *3*)
Negative ::= NEG
Preposition ::= PREP=(*on* | *to* | *over* | *under* ...)
Tense ::= TENSE=(*past* | *future* | *prog*)
Perfect ::= PERF
Progressive ::= PROG
Passive ::= PASSIVE
Modal ::= MODAL= (*could* | *will* | *would* ...)
Form ::= FORM= (*pre* | *post* | *preaux* | *postaux* | ...)
Infinitive ::= INF

The *Id* is a unique identifier generated by the framework. It is used to reference an element, and can appear as an attribute value on certain referring attributes.

3.7 A Practical Implementation

The CAMEO language is a relation defined between surface text and a computational representation. An initial stage of processing is required to transform the raw surface text into the internal form of the representation. In this section I will report on an implementation of the CAMEO language, and the corresponding processing that illustrates how text is transformed into the representation.

The CAMEO representation language is designed to support the encoding of multiple levels of linguistic analysis from a wide range of sources. Each source requires an independent processing module to transform its output into the CAMEO language (unless the source supports the CAMEO representation internally). For example, the initial implementation of CAMEO was developed using the Link Grammar Parser (Sleator and Temperly, 1993) by creating an independent processing module to transform the Link Grammar output into the CAMEO representation. The current implementation, which I will be discussing at length in this section, transforms the output of the probabilistic parser included in the RASP suite of text processing tools (Briscoe and Carroll, 2002). Although this transformation is an independent process, the RASP parser was a crucial part of the development of the CAMEO representation. Additionally, many constructions in the representation necessarily follow the analyses of the parser. For these reasons, in the next section I will first introduce the syntactic processing performed by the RASP suite. (I used several versions of the RASP tools over the course of this work, ranging from Version 2 with tsg12 through Version 3.1 with tsg15; however the syntactic processing is similar across all versions).

3.7.1 RASP Syntactic Processing

The RASP toolkit processes text serially through a series of modules, ultimately producing a set of statistically ranked deep parses. For the purposes of developing the CAMEO representation, multiple parses were not considered and the highest ranked parse was selected in each case. The following steps describe the processing of the individual modules in the RASP toolkit (see Briscoe and Carroll, 2002 for full details):

1 The first stage is tokenization of the raw text using a deterministic finite-state transducer. This includes deciding word and sentence boundaries in the context of white space and punctuation. The sentence boundaries determined by RASP are used as the sentential contexts in the CAMEO representation (see Section 3.5.2.1 above).

2 The next stage is a statistical tagger which assigns PoS and punctuation tags to individual words. The tagger is implemented using a HMM and assigns probabilities to each tag for ranking purposes. A configurable threshold is used to select tags to be included for processing in the following stage.

3 This stage performs a (deterministic) morphological analysis on each of the word+tag pairs, resulting in a lemma+morphological suffix based on the word and its PoS tag.

4 This stage parses the multi-tag lattice using a manually-constructed grammar of PoS and punctuation tags.

5 Finally, the individual parses are assigned a probabilistic ranking based on the syntactic analysis and available lexical information.

The RASP toolkit is a flexible system that is capable of generating several output formats. Because the CAMEO representation attempts to represent as much of the linguistic information as possible, the transformation module was developed to use the detailed syntactic parse tree output of RASP. Figure 3.7.1 shows a sample of the parse tree format produced by RASP. Each phrase of the parse is represented by a node in the tree labelled with a phrasal category followed by information internal to the parser. Words of a sentence are shown as lemmas concatenated with their corresponding PoS. For example, the prepositional phrase in Figure 3.7.1 begins with the node labelled |PP/p1|, followed by the preposition [*in*] represented by |in_II|.

The top level node is labeled with [T], and a successful parse of a sentence will be labeled with [S]. A partial parse will be returned where a complete parse cannot be found, consisting of a sequence of parses covering the input.

```
(|T/txt-sc1/----|
      (|S/np_vp| (|NP/n1_n1-name/-|
            (|N1/n| |Oscar_NP1|))
            (|V1/modal_bse/-| |should_VM|
                  (|V1/be_pp/--| |be_VB0|
                  (|PP/p1|
                     (|P1/p_np| |in_II|
                     (|NP/det_n1| |the_AT|
                        (|N1/n_n1/-| |engine_NN1|
                        (|N1/n| |room_NN1|))))))))))
```

Figure 3.7.1 – RASP syntactic parse tree output for [*Oscar should be in the engine room*]

3.7.2 Transformation

The transformation from the RASP parse tree to the CAMEO representation is computed using an independent transformation module, customized for the specific

syntactic output of the RASP system. Different parsers would require different transformation modules to be developed, however all transformations from equivalent syntactic parses should result in equivalent representations.

The current transformation is implemented using an event-driven tree-walking approach. The parse tree is traversed in-order, and each node in the tree initiates an event. Each event generates some context, element, or attribute in the representation, depending on the syntactic constituent of the node. For example, when the transformer encounters an NP a new **obj** element is created, a VP initiates an **evt** element, and so on. Within a syntactic node, the PoS tags are used to further drive the process. The result is a deterministic transformation of the syntactic parse tree into the CAMEO representation.

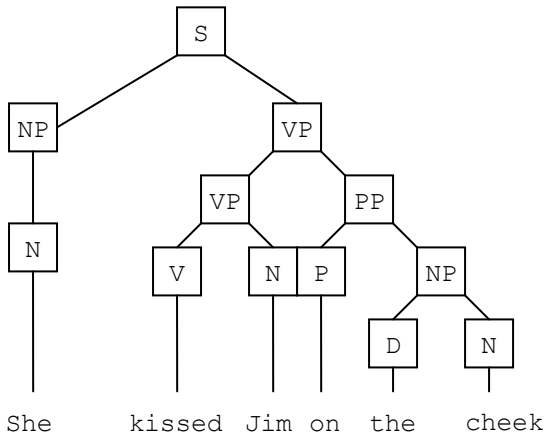
Although this approach would normally be considered equivalent to a compositional approach, there are several cases which require special processing. For example, when transforming compound noun phrases, the number of noun **obj**s must be discovered beforehand so that constituent NP events create **obj** elements properly within the group **obj** container (see Section 2.2). The representation of possessive nouns requires similar treatment. The transformation of the VP into the **evt** element is another special case, which requires the entire structure of the VP to be known before certain attributes can be created (e.g. the passive construction).

Contexts are created based on the parser's interpretation of a clause. Whenever the parser marks a syntactic node as a sentence or clause, a new **ctx** element is created. These contexts in turn become containers for the various elements transformed from the syntactic constituents of the clause. Thus, the various constituent elements (**obj**, **evt**, etc.) are relative to the clausal context they are found in.

As I mentioned above, the PoS tags are used to derive certain element attributes in the CAMEO representation. Combined with the lexemes, these allow a limited amount of interpretation for extended linguistic features using sure-fire rules. Pronouns generate appropriate attributes for animacy, gender, number, and person. Certain style nouns (e.g. *Mr.* and *Mrs.*) generate gender and/or animacy attributes. Other attributes are computed using the syntactic context, such as the form attribute which encodes the position of e.g. adverbs relative to a phrase head. Note these attributes are a

computational convenience for subsequent processors and do not denote any semantic meaning, since the system is not capable of determining metaphoric and other meta-linguistic usage.

Figure 3.7.2 shows a graphical depiction of a syntactic parse tree and the resulting transformation into the CAMEO representation. The top level [S] generates the outer **ctx** element, which serves as the contextual container for all descendent constituents. The subject NP [*She*] produces an **obj** element with attributes indicating the pronoun lexeme, female gender [G=*f*], and animacy [A]. Next, the VP is traversed in order. The head verb [*kissed*] is discovered and stored. Then the direct object [*Jim*] produces another **obj** element containing a **name** element. The PP phrase is then visited generating a **rel** element container. Because the verb **evt** element has not yet been created, the prepositional **rel** element is left floating temporarily. The preposition [*on*] is recorded as an attribute and the prepositional object is processed resulting in a third **obj** element being created. Once the entire VP has been visited, the **evt** element is created. The appropriate object references are set for the subject [S] and direct object [O] attributes. The head verb and tense attributes are then filled in. Finally the prepositional **rel** element is inserted in the verb **evt** element.




```
ctx [ TYPE=clause
  obj[ ID=o1 PRON=she G=f A=h]
  obj[ name[ Jim ] ]
  obj[ ID=o3 DET=the class[cheek]]
  evt[ S=o1 ACTION=kiss
       TENSE=past O=o2
       rel[ PREP=on OBJ=o3 ]]]
```

Figure 3.7.2 – Graphic depiction of a syntactic parse tree and corresponding CAMEO representation

3.8 Conclusion

In this chapter I have described the CAMEO text representation language which forms the experimental framework used in the remainder of the thesis. The representation was designed to satisfy the desiderata proposed in the introduction, and extends equivalent extant text representation strategies to include a systematic treatment of the representation of context, as well as other innovations. Central to the organization of the representation is the concept of contexts, which are containers for the semantic and pragmatic information about the document. The contexts are used during processing to define semantic scope for modules that can take advantage of it. QA and coreference resolution were given as examples of tasks that can be enhanced with contextual information.

The current implementation of the CAMEO language was developed using the RASP system. The output of the RASP syntactic parse is transformed through an independent process into the CAMEO representation. Alternate linguistic technologies will have different output representations that require corresponding transformation modules to be developed.

4

Operations: Realisation and Manipulation

In this chapter I will present two key operations which are enabled by the properties of the representation. The extent to which CAMEO enables these operations (and their implementations) is another differentiator of the CAMEO representation.

In Section 4.1 I will discuss surface realisation and the process of transforming a CAMEO representation back into surface text. I will explain how the structure of the representation is used to realise objects, phrases, and sentences. I will then present an experiment designed to evaluate some of the surface realisation capabilities of the representation.

In Section 4.2 I will discuss how the representation can be programmatically manipulated to change the form and meaning. I will explain how CAMEO has advantages over other representational forms for this type of operation. I will then contrast manipulations using CAMEO with other representations on two experimental tasks.

4.1 Surface Realisation

Natural language generation is sometimes modelled having three components (Elhadad and Robin, 1996). At the highest level is *macroplanning*, which addresses the overall content and structure of the output. Macroplanners attempt to satisfy some

communicative goal, by selecting appropriate information and determining appropriate rhetorical structures. Macroplanners operate at much higher semantic levels than the CAMEO representation supports directly. Macroplanners typically manipulate meaning representations at the level of paragraphs or groups of sentences, and could use the structural context elements in CAMEO to direct the document level output. However, in this section I will focus on the sentence level realisation, and thus macroplanners are not applicable to the discussion at hand.

The next level of generation occurs with *microplanning*. This level of planning addresses sentence-level meaning representations. Microplanners decide issues such as sentence form, referring expressions (including anaphora), lexical choice, and amalgamation (Hovy, 2000). The microplanner produces symbolic representations describing the sentence to be generated. At this level all of the strategic choices for the textual output have been made.

The final stage of generation, *realisation*, involves the linguistic expression of the symbolic representations produced by the microplanner. The surface realisation component is tasked with interpreting the lexical content according to any syntactic parameters to produce syntactically and grammatically well-formed text. Implementations of surface realisers typically integrate a grammar component along with some means for specifying surface variations (e.g. Elhadad and Robin, 1996).

The CAMEO language is an intermediate text representation that is suitable for integration with a surface realiser. The CAMEO representation abstracts a subset of lower-level grammatical functions without committing to a grammatical (or semantic) theory. This approach allows CAMEO to provide the same intermediate role to surface realisers as for semantic processing modules. That is, the fine-grained text representation and surface feature parameters of the CAMEO language can serve as a common interface to different surface realisation systems, allowing the lowest levels of linguistic expression to be abstracted.

Because many text processing tasks (including some attempted in this thesis) require the realisation of the output as surface text, as part of the CAMEO framework, I have developed a transformation which deterministically maps the CAMEO representation to surface text.

The realisation of surface text from the internal representation is accomplished by essentially running the transformation described in Section 3.7.2 in reverse. Instead of traversing the syntactic parse tree, the surface realisation processes the representation using a similar event-driven approach. Like the syntactic nodes in the transformation of surface text, elements in the representation initiate events which generate the surface text expression.

Figure 4.1 gives a high-level pseudocode for the general surface realization algorithm. The transformation into surface text is driven by the primary types defined in the model: **ctx**, **obj**, and **evt** elements. The algorithm processes elements recursively in order as shown in the pseudocode. This basic algorithm will successively build up constituents, phrases, sentences, paragraphs, and ultimately documents. It is a deterministic transformation that is essentially compositional, and it is currently implemented using the XSLT transformation language. (Wilcock (2001) discusses the advantages of using XML/XSLT in the context of a pipelined NLG system, however, the representations used at each stage are somewhat task-specific and do not attempt to provide a general text representation as developed here).

The top level context (i.e. a **ctx** element of type document) begins the transformation process. Given a **ctx** element, the generator first attempts to transform all **evt** elements (not referenced by other elements) in the order they appear. Events will usually include references to objects and/or contexts that the generator processes recursively. After transforming all events contained within a context, or if no events occur, the generator will then attempt to transform any other elements (e.g. **obj**s, **rel**s, etc.), in the order they appear.

Each element is transformed using its attributes to determine the lexical expression. For example, an **evt** element has an action attribute which encodes the head verb stem. The transformation algorithm looks up the verb stem (in the **lexis** context), and passes it to a morphological processor along with appropriate features (e.g. person and tense). The output is the fully realized surface form of the word. The representation language includes special attributes on **ctx**, **obj**, and other elements, which specify certain surface variations. Thus it is expressive enough to facilitate multiple surface forms of a representation through manipulation of these features.

```
begin
   for each context of type document
      for each context of type sentence
         for each element not referenced by another element
            generate element
end

begin generate ctx type clause
   if conjunction then print conjunction
   for each element not referenced by another element
      generate element
end

begin generate evt
   if passive generate object obj else generate subject obj
   for each modal
      generate modal
   if perfect generate perfect
   if participle generate participle
   if neg print "not"
   if progressive generate progressive
   generate head verb
   if passive generate subject else generate object
   for each element of type mod rel inf ctx
      generate element
end

begin generate obj
   if determiner generate determiner
   if quantifier generate quantifier
   if this obj possessed generate possessor obj
   if class generate class
   if pronoun generate pronoun using proper case
   if name generate name
end

begin generate mod
   if quantifier generate quantifier
   if negative print "not"
   print lex
   for each inf
      generate inf
end

begin generate rel
   print preposition
   for each obj
      generate obj
end

begin generate inf
   with EVT reference generate evt
end
```

Figure 4.1 – Pseudocode for surface realization from CAMEO representation

As an example of generating from a semantic representation, consider the representation for the sentence [*The doctor examined a patient*], shown in (74).

(74)
ctx
[TYPE=*sentence*
obj[ID=*o1* DET=*the* **class**[*doctor*]]
obj[ID=*o2* DET=*a* **class**[*patient*]]
evt[S=*o1* ACTION=*examine* SO=*o2* TENSE=*past*]
]

The transformation would begin processing the **ctx** element, by recursing into the container. Within the container, the next element to process would be the **evt** element (because the **obj** elements are referenced by another element). According to the transformation algorithm, the **evt** element would be processed by first generating the subject **id**(*o1*), next the verb, and finally **id**(*o2*).

To generate **id**(*o1*), the algorithm first processes the determiner, followed by the class elements in order. Because **id**(*o1*) has no plural attribute, and is not a group container, each word is transferred directly from the lexis, producing the surface text: [*the doctor*].

To generate the verb, the algorithm reads the tense attribute, determines the person of the subject **id**(*o1*), and passes these along with the head verb stem (from the ACTION attribute) to the morphological processor, producing the surface text: [*examined*]. The algorithm then processes **id**(*o2*) similarly to **id**(*o2*), producing the surface text: [*a patient*].

After fully processing the **evt** element, the transformation algorithm would then examine the remaining objects in the container (**id**(*o1*) and **id**(*o2*)). In this case, these are ignored because they were previously referenced by another element (i.e. the **evt** element). Because the type attribute of the **ctx** element container has a value of *sentence*, the resulting surface text is processed as a sentence (i.e. end-punctuation is added using deterministic rules). The final output becomes: [*The doctor examined a patient.*].

4.1.1 Generating Object References

The transformation algorithm will normally use all existing properties of an object to generate a reference. For example, the surface realization of the object in example (75) is [*the clothes drying on the line*].

(75) **evt**[ID=*e1* ACTION=*dry* PROG INF **rel**[on the line]]
obj[ID=*o2* DET=*the* **class**[clothes] **inf**[IDREF=*e1*]]

However, in natural language an object may be referred to in a variety of ways throughout a document. For instance, **id**(*o2*) may be referred to as [*the clothes drying on the line*], *[the clothes drying*], [*the clothes on the line*], [*the clothes*], or even [*they*]. Linguistic phenomena such as anaphora, determiners, deterministic adjectives and demonstratives are used extensively in natural language, complicating the task of generating referring expressions. Deciding and planning these references is deferred to other modules because the CAMEO representation is designed to be theory-neutral. However, a processing module is able to control the form of the referring expressions by manipulating the representation.

The method for controlling the expression of object references is to create a new object with the desired attributes, making it equivalent with the object referred to. For example, if there is an object element **obj**[ID=*o1* **name**[*John*]], and the text planning module decides to use a pronoun reference, then a new object **obj**[ID=*o2* PRON=*he* EQ=*o1*] can be created, and the old object moved out of the processing context. The surface transformation will process **id**(*o2*), generating the surface text *he* , but semantic processing will see the two objects as equivalent, due to the EQ attribute on **id**(*o2*).

4.1.2 Generating Phrases and Sentences

As explained in Section 3.5.2.1, a phrase is represented in the model using a **ctx** element. Contexts usually include more than a simple object, however they do not necessarily represent a complete sentence. Contexts can be referenced in event structures as phrasal or sentential complements, or they can be unreferenced, e.g. when participating in a conjunct.

A context is transformed into a surface text phrase using the algorithm described above. The type attribute on a **ctx** element determines whether a context is a phrase or a sentence. If the context is a phrase, no further processing is required. The containing or referring context generates the necessary punctuation.

The transformation into complete sentences is an extension of generating phrases. The surface generator produces a complete sentence for every context that has a sentence attribute set. Normally, a discourse context is composed of a number of sequential sentence contexts (these may be grouped in higher level contexts such as paragraphs). By processing these in order, the generator produces the output document.

If a sentence is to be formed using the conjunction of two or more phrases, the first phrasal context must contain the other contexts. The containing context will have a conjunction attribute set that the generator uses to determine the sentence form. For example, the representation in (76) generates the following surface text: [*Tom pulled the brake, but the train did not stop*].

(76)

ctx [TYPE=*sentence* CONJ=*but*
  obj[ID=*o1* **name**[*Tom*]]
  obj[ID=*o2* DET=*the* **class**[*brake*]]
  evt[S=*o1* ACTION=*pull* O=*o2* TENSE=*past*]
  ctx [
    obj[ID=*o1* DET=*the* **class**[*train*]]
    evt[S=*o1* ACTION=*stop* TENSE=*past* NEG]]]

Contexts are processed recursively, and the containing context determines the form an individual phrase takes. In example (76) the containing context carries the CONJ attribute, which determines the form of the second phrase, i.e. conjunctive clause.

Other sophisticated constructions are possible using this approach. For instance, subordinate clauses, appositives and quoted speech. The representation is designed to unambiguously represent all possible surface forms, giving a discourse planning module complete control over the generated output.

4.1.3 Experiments

In this section I present the results of experiments designed to systematically test both the surface realisation transformation, and the expressiveness of the CAMEO language as a text representation. Examples of the wide range of surface forms available in English text are given in (77), which all yield practically identical semantic interpretations.

(77) *Nancy gave the book to Tom.*
Nancy gave Tom the book.
Nancy to Tom gave the book.
To Tom Nancy gave the book.
Tom, Nancy gave the book to.
The book was given Tom by Nancy.
The book was given to Tom by Nancy.
The book was given by Nancy to Tom.
The book, Nancy gave Tom.
The book, Nancy gave to Tom.
The book, to Tom Nancy gave.

Some of these variations are obviously less natural than others, but they help illustrate the point that there can be a wide variety of surface syntactic variations for a sentence, even if the semantic representation is unambiguous.

Selecting among the surface variations is the job of microplanning modules as mentioned earlier. The CAMEO language provides a means to encode these selections by using special attributes on the elements of the representation. Some of these attributes have already been mentioned (e.g. `PASSIVE` in Section 3.4.6), others will be discussed below. These attributes give microplanning modules the flexibility to specify all forms of surface realisation, while being insulated from the mechanics of the transformation.

In order to evaluate the CAMEO language for adequacy in expressing surface variation, a wide range of surface forms is needed. Since this experiment is not intended to test microplanning or other high-level text generation components, a systematic means of producing instances of the representation was used instead of external text

generation methods. Figure 4.1.3 shows a block diagram of the flow of processing in a typical text generation application, and the alternative flow used in these experiments.

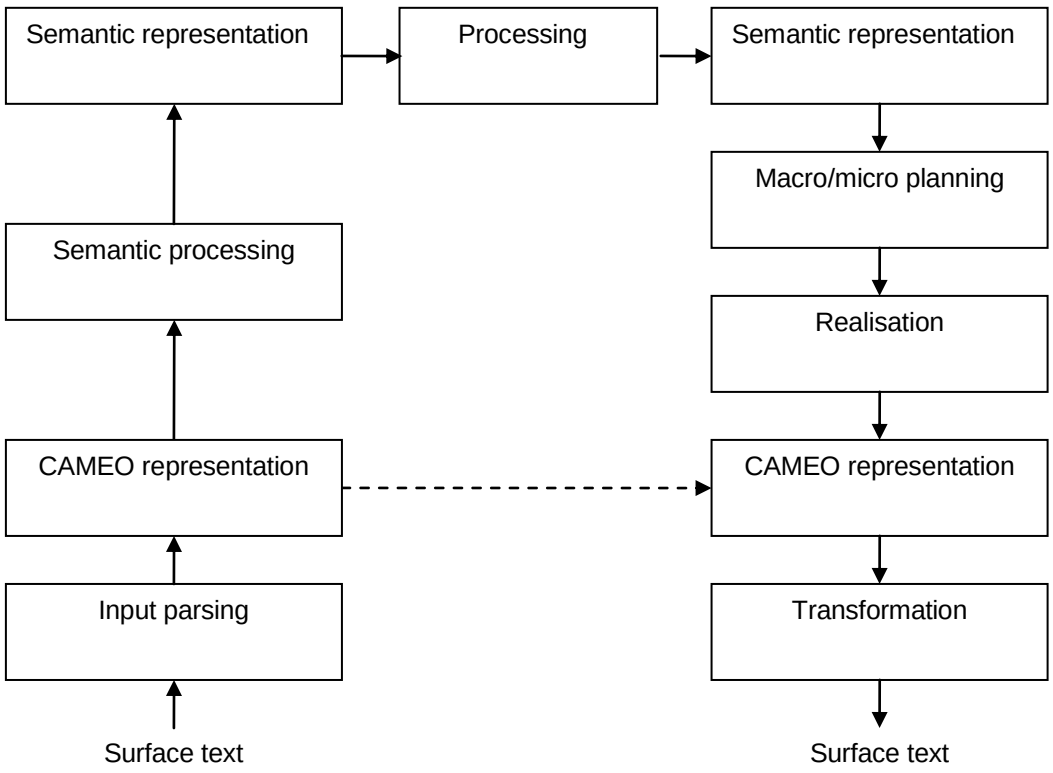


Figure 4.1.3 – Block diagram showing flow of typical text generation application. Dotted line shows alternate flow of experiments.

The experiment consists of a set of sentences encoded in the CAMEO representation as described in Section 3.7. This representation is then fed back through the surface realisation transformation described in Section 4.1 and the output is compared with the original source text. This effectively simulates processing under a real application, while providing an objective measure of evaluation.

A completely comprehensive test of available surface forms is impractical. However the sentences used in the experiment include a wide sample of surface variation available in the English language. The test sentences are from the development platform of the RASP parser (Briscoe and Carroll, 2002), and thus have been used to evaluate various parser grammars.

There are four sets of test sentences, each with a different general emphasis:

Set 1 This set is comprised of mainly verb phrase variants including various prepositional phrasal constructions. It includes a sampling of tenses, e.g. [*Kim will have been abdicating*], as well as verb particles [*Kim made off with the butter*], and complements [*That he would apologize was clear to me*].

Set 2 This set contains mostly noun phrase variants, especially wh-gaps and dependent modifying clauses. It also includes quantifier variations [*All of the butter melted, All the abbots came*], and possessives [*The abbot's many abbeys closed*].

Set 3 This set has less common syntactic constructions such as object and prepositional fronting [*At Sandy Kim acknowledges that Lee looked*]. It also includes relative clauses [*The abbot about whom Kim has some doubts is crazy*] and contractions [*Kim can't not help Lee*].

Set 4 This set includes sentences testing more complex relative clauses [*Kim is abandoning Lee now when he is eager to be helped.*] including certain pronouns [*Those in the abbey who abdicated were crazy*] and quantifiers [*The half that Kim helped are crazy*].

4.1.3.1 Results

A quantitative score is not very helpful in this case. However, Table 4.1.3.1 shows some statistics about the error rate of the system. These numbers are listed for each level of the evaluation. The number of sentences used in a particular test is recorded in the second column of the table.

Table 4.1.3.1
Error rates for surface transformation

		Errors			
Test	**Sentences**	**Constituent**	**Punctuation**	**Total**	**Syntax**
1	209	0	0	0	0
2	152	3	2	5	0
3	191	13	1	14	0
4	33	7	14	21	0
Total	585	23	17	40	0

The error rates were calculated by comparing the input text with the output of the text transformation from the representation. Parse errors (judged from the most probable parse returned by the parser) accounted for 45 sentences and were discarded. An error was recorded if the output text did not match the input text exactly, including punctuation but excluding whitespace. The type of error was manually classified according to syntax, constituent, or punctuation according to the following criteria:

A syntax error occurs when the output sentence is not arranged in the same manner as the input, e.g. conjunctive clauses appearing in the incorrect order.

A constituent error is recorded when a member of a noun or verb phrase is incorrectly realised. This could be, for example, the incorrect ordering of modifiers or prepositional phrases. It can also signify the omission of a constituent such as the indirect object of a verb.

A punctuation error is recorded if incorrect punctuation is used or omitted, e.g. commas marking appositives, or the trailing punctuation on a sentence.

4.1.3.2 Analysis

The CAMEO representation and surface transformation was able to correctly regenerate a majority (93%) of the test sentences. There were no syntactic errors due to the surface transformation. All of the constituent errors reported in Table 4.1.3.1 were due to the conversion of the parse trees into the CAMEO representations. The majority of these errors were due to erroneous or missing gap analysis. For example, sentence (78) is from Test Set 3 and shows an example of a prepositional gap. Note that the derived CAMEO representation did not assign [*Kim*] as the object of the infinitival

complement of [*desire*]. The RASP system used for these experiements does not explicitly indicate or coordinate gaps on the parse tree output, so gaps and control must be inferred by the transformation process which converts the parser output into the representation. In this and several similar cases, the fronted object was not interpreted correctly and instead left unconnected to the phrase. Although this coincidentally resulted in the correct surface expression (disregarding punctuation), it was recorded as a constituent error because the representation was incorrect.

(78) *Kim, Lee has a crazy desire to help.*

obj[ID=*o1* **name**[*Kim*]]
obj[ID=*o2* **name**[*Lee*]]
obj[ID=*o3* DET=*a* **mod**[*crazy*] **class**[*desire*] **evt**[ACTION=*help* INF]
evt[S=*o2* ACTION=*have* TENSE=*present* O=*o3*]

As a further validation of the surface realisation, the erroneous representations for each of the constituent errors were hand-corrected and the subsequent surface realisations proved correct.

Punctuation comprised the smallest percentage of the errors observed in the experiments. In fact, all the recorded punctuation errors were the result of spurious or missing commas related to clauses and preposed NPs. The surface realisation of commas is currently driven by phrasal context elements and in some cases this proved insufficient. Other types of errors such as end punctuation did not occur, however a systematic test of punctuation was not included in this experiment.

4.1.4 Discussion

One of the goals of this section has been to demonstrate the potential for surface text realisation using the CAMEO representation language. Using CAMEO as a surface description abstracts the mechanics of realisation from a text generation system, allowing it to generate and manipulate the CAMEO language instead of internal task-specific forms. This approach would facilitate the integration of multiple text generation technologies, providing a common, computationally tractable form for comparison and test.

Because CAMEO is positioned near the surface syntactic level it does not impose a grammar on the representation, and can thus support microplanners with a range of grammatical approaches. This flexibility precludes the representation from abstracting lexical issues such as light verbs and verb particles, i.e. the representation enforces no restrictions on improper usage of these types of constructions. Thus a microplanner would likely need to incorporate some level of lexical information to construct CAMEO representations which realise well-formed surface expressions.

Besides testing the surface realisation component of CAMEO the experiments exposed another possible application of surface realisation. All of the non-punctuation errors in the experiment (after discarding parser errors) were the result of errors in the conversion to the CAMEO representation, and not due to flaws in the surface realisation. Therefore, running text through a process similar to that demonstrated in these experiments can potentially serve as a test for successful conversion. That is, text that has been converted into the CAMEO representation and realised as surface text should exactly match the original source (disregarding parse and punctuation errors). If it doesn't, it most likely indicates an error in the conversion to the CAMEO representation.

These experiments were further intended to serve as an informal evaluation of the expressiveness of the representation itself. While not comprehensive by any means, the tests did include a sample of the syntactic variation encountered in English. The representation achieved a 93% success rate on the test set, and all the errors resulted from either the transformation of the parse trees into the CAMEO representation, or minor punctuation errors. The internal properties and attributes built into the representation proved expressive enough to realise the wide range of syntactic formations used in the experiment, while still supporting a generally recursive and deterministic transformation.

4.2 Text Manipulation

The intrinsic object-orientation of the CAMEO representation is compatible with programming paradigms of technologies such as DHTML and XML. This simplifies manipulation of the internal representation, and when combined with the surface

realisation component, provides a tool for generating surface variation. Modifications to the internal representation are accomplished by changing the value of attributes, or adding/deleting/moving elements. For example, a program could convert all sentences in a document to the passive/active voice by changing the passive attribute on all finite event objects. Text can be condensed by deleting relative clauses, which only involves searching for and removing certain context elements. Such operations require no linguistic knowledge, and are basic functions of DOM processing.

In this section I will attempt to demonstrate the advantages of using the CAMEO representation for text manipulation over traditional transformation-based approaches. In Section 4.2.1 I will give a qualitative comparison of CAMEO with other approaches on several operations related to sentence condensation. In Section 4.2.2 I will present the results of experiments on sentence activsation and give a comparative analysis with an extant system to show that CAMEO can produce equivalent or better alternations with simpler manipulations and a more flexible framework.

Text manipulation is typically implemented in the literature using rule-based transformations. A set of rules are applied to a representation and those rules that match are executed to produce the altered output. For example, Riezler et al. (2003) describe a system for sentence condensation using rules of the form

```
+adjunct(X,Y),in-set(Z,Y) ?=> delete-node(Z,r1), rule-trace(r1,del(Z,X))
```

applied over LFG f-structures. The left side of the rule is used to match portions of the representation, and the right side of the rule specifies a transformation.

Another example of a rule-based framework for manipulation is SYSTAR (Canning, 2002). SYSTAR is an automated system intended for text simplification which manipulates several syntactic constructs. One construct which is addressed in detail is the activisation of passive verb phrases. For example, a rule which transforms a simple passive construction followed by a comma is

```
((VPPAS(?b)PP("by" II)NP(?c)("," |,|)(?Y)))=>((?a)(?c)(?b)(?X)(?Y))
```

The rules are applied to the dependency information output from the RASP syntactic parser.

In many rule-based systems such as these, the rules are made to be as general as possible in order to achieve the widest application. However, it usually requires several variations to achieve adequate coverage. For example, SYSTAR contains 17 rules to cover the active/passive transformation. As the system is developed and new cases are discovered which fall outside the scope of the system, new rules must be added to address them.

The CAMEO representation in effect inserts an intermediate step in the rule-based approach. The normalised text representation provided by CAMEO can be thought of as storing the results of much of the left side matching that a rule-based system would do, and the realisation component of CAMEO is functionally equivalent to the right side of a rule-based system.

For example, passive verb phrases are recognised (i.e. 'matched') when syntactic information is transformed into the CAMEO representation. The `PASSIVE` attribute is attached to verb phrases which are expressed in the passive form. This encodes the more complex part of a passive match rule in the simple attribute `PASSIVE`. Subsequently, a system which needs to find passive verb phrases need only search for the `PASSIVE` attribute, rather than a series of complex dependency patterns.

The surface realisation, which resembles the right side operations of a rule-based system, is driven by the objectified attributes and elements in the CAMEO representation. Changes to these elements and attribute values direct the form the surface text will have. So again, rather than relying on a complex dependency pattern match to fire a rule's right hand transformation, the simplified encoding of attribute and elements serves to enable various functions in the surface realisation component.

The conversion of linguistic information into the CAMEO representation language essentially decouples the rules, and is one advantage of the CAMEO model over rule-based systems. In rule-based systems, the left hand matching expression is tied directly to the right hand transformation. If there are two or more rules which use the same match or transformation, parts of them must be duplicated. In contrast, CAMEO allows the rule matching, expressed as the encoding of the representation language, to be developed independently of the right hand transformations (surface realisation function).

For example, two different parsers were used during the course of this research. Transformations from the parser syntactic output into the CAMEO representation were developed for both parsers. This is analogous to using two different sets of 'matching' rules with a common, normalised intermediate representation. Once encoded into the CAMEO representation, however, manipulations can be processed identically and the same set of 'transforming' rules can be applied (i.e. the surface realisation function).

Another advantage of the CAMEO model is the manipulations are reversible, i.e. manipulations which are driven from attribute values can simply be reverted. For example, removing the `PASSIVE` attribute will activise a verb phrase, while adding the `PASSIVE` attribute will make it passive again. Verb tense and aspect can also be manipulated in this way. In contrast, rule-based systems produce altered forms of their input which would require a separate set of rules to revert. For example, the SYSTAR rule shown above will produce an altered dependency list with an activised verb phrase. However, the rule will no longer apply to this new output, requiring a complementary rule to be created to revert to the original form.

Finally, the CAMEO model allows multiple simultaneous manipulations. For instance, several properties of a verb phrase can be changed by changing various attribute values on the **evt** element. This could, say, change the tense, aspect, and activise the sentence all at once. The text only needs to be realised after the changes have been made. In a rule based system, several stages may be necessary to achieve the same result. For example, one rule may transform part of the dependency information another rule requires to match. The transformation must first be carried through before submitting to the next rule, which is less efficient and further complicates reverting to the original.

4.2.1 Sentence Condensation

Sentence condensation is an application of text manipulation that attempts to shorten sentences while keeping the essential meaning. Linguistic tasks such as text summarisation and simplification typically use techniques derived from sentence condensation (Knight and Marcu, 2000).

The CAMEO representation provides a simple framework for these types of manipulations. Because constituents are explicitly encoded and objectified as elements, removing them can be accomplished using very simple operations. For example, prepositional phrases can be removed by deleting **rel** elements, adjectives can be removed by deleting **mod** elements, and clauses can be removed by deleting **ctx** elements.

Tasks that perform manipulations on a text representation must first have some means of locating specific constructions. Representations that include deep syntactic relations, such as CAMEO, are well suited for this stage of processing because the more complex relations have been pre-analysed and encoded in the representation. This contrasts with more shallow representations, where complex pattern-based rule transformations must be used because dependency information is not encoded in the representation. Requiring rules to include deep syntactic patterns complicates the process of locating the correct constituent to condense. Excising the constituent and reforming the sentence can be difficult as well for the same reason.

For example, Chandrasekar et al. (1996) report on sentence simplification using a finite state grammar over tagged text. Their system used rules of the form

```
X:NP, RelPron Y, Z -> X:NP Z. X:NP Y.
```

As reported in their findings, the shallowness of the syntactic information limits the effectiveness of this approach. Without richer dependency information certain constructions such as long-distance dependencies and ambiguous attachments cannot be handled properly.

Riezler et al. (2003) report on sentence condensation using LFG, which includes deep syntactic dependency information. Like CAMEO, LFG encodes adjuncts directly and is therefore able to remove certain constituents with simple rules. The example at the beginning of the chapter, also from Riezler et al., gives a simple rule for deleting adjuncts by searching f-structures for nodes belonging to the adjunct function. The rule's right side specifies the operation to perform on the f-structure.

Sentence simplification using LFG consists of applying a set of these reducing rules to the f-structures. Once the f-structures have been modified, a realisation stage is run over the f-structures to generate output. Because the rules are independent of the f-structure grammar, they are not guaranteed to produce valid f-structures, so the grammar in the generation stage serves as a filter to test for well-formed f-structures. F-structures that pass this filter are used to produce surface text (Riezler et al., 2003).

Because the realisation of an f-structure is non-deterministic, a single f-structure may produce multiple surface strings, and multiple strings may correspond to a single f-structure. This is a disadvantage when developing transformational rules because it becomes more difficult to measure a rule's effect on the output when it is non-deterministic. A loose correlation between a rule and the system's performance will degrade automated approaches to rule learning, and will tend to result in redundant and larger sets of rules.

While it is desirable to produce grammatical output, for tasks that require regeneration the use of bi-directional grammars like those used in LFG has some disadvantages. These grammars are often complex, hand-crafted resources developed over long periods of time, making them difficult to maintain or extend for new domains. For certain tasks, this level of sophistication may be overkill, and it may be more desirable to have a simple regeneration component that can realise surface text directly from the representation.

The CAMEO representation is well-suited to the sentence simplification/condensation task. Like LFG, CAMEO includes explicit encoding of constituents and thus supports simplistic transfer rules. However, CAMEO uses a deterministic realisation component which produces a single surface expression for each representation. This avoids the ambiguity in rule evaluation and development.

CAMEO is also not constrained by a grammar theory, and supports robust and partial processing methods. This allows condensation of individual constituents which are not necessarily part of a well-formed sentence. For example, objects in a list or table could be simplified using rules that operate on **obj** elements which are unconnected to **evt** elements. The deterministic surface realiser is able to generate individual elements or fragments in cases such as these.

Another advantage the CAMEO representation provides over extant sentence simplification approaches is the representation of context. The recursive contextual structure employed in the CAMEO representation language allows condensation operations beyond the sentence and phrasal levels. For example, larger structural constituents like sections and chapters can be removed using the same operations described above. Simple condensation techniques such as removing all but the first sentence in each paragraph can also be accomplished with similar transformational rules. Having a single representation which facilitates transformations at all levels allows integrated approaches that would otherwise not be possible using sentential-based representations.

To demonstrate some of the transformations possible using the CAMEO representation, I will compare rules in the CAMEO system with rules in other systems for several sentences reported in the literature. I will report the CAMEO rules in pseudo-code and give examples of actual XSLT processing instructions which operate over the CAMEO XML implementation.

Riezler et al. (2003) list a set of LFG condensation rules to transform sentence (79). The reported condensed sentence is shown in (80).

(79) *A prototype is ready for testing , and Leary hopes to set requirements for a full system by the end of the year*

(80) *A prototype is ready*

The LFG transfer rules reported to produce this sentence are reproduced in Figure 4.2.1-1 below. Each line represents an operation on the f-structure and the resulting transformation is used in subsequent rules.

```
rtrace(r13,keep(var(98),of)),
rtrace(r161,keep(system,var(85))),
rtrace(r1,del(var(91),set,by)),
rtrace(r1,del(var(53),be,for)),
rtrace(r20,equal(var(1),and)),
rtrace(r20,equal(var(2),and)),
rtrace(r2,del(var(1),hope,and)),
rtrace(r22,delb(var(0),and)).
```

Figure 4.2.1-1

Example of LFG transformations for sentence condensation

ctx [ID=*t1* TYPE=*sentence*
obj[ID=*o1* DET=*a* **class**[*prototype*]]
obj[ID=*o2* **class**[*testing*]]
evt[ID=*e1* S=*o1* ACTION=*be* TENSE=*present* **mod**[*ready* **rel**[*for testing*]]]
ctx [ID=*t2* CONJ=*and*
obj[ID=*o3* **name**[*Leary*]]
obj[ID=*o4* **class**[*requirements*]
rel[*for* **obj**[DET=*a* **mod**[*full*] **class**[*system*]]]]
evt[S=*o3* ACTION=*hope* TENSE=*present*]
evt[S=*o3* ACTION=*set* INF
rel[*by* **obj**[DET=*the* **class**[*end*]
rel[*of* **obj**[DET=*the* **class**[*year*]]]]]]]]]

Figure 4.2.1-2

CAMEO representation for sentence (79)

The CAMEO representation of the full sentence is given in Figure 4.2.1-2. The corresponding operations to produce the condensed sentence from the CAMEO representation are show in Figure 4.2.1-3. For each transformation, an abstract pseudocode is given, followed by the actual XPath expression that would be applied to the representation. The transformation is implemented in XSLT using an identity transform template (which copies elements unaltered), augmented with the specific rules to be applied. In the case of deletion, the null template is used which does not copy any node matched by the expression, including all descendent nodes.

Pseudocode: delete **mod**[*ready*] in **evt**[*e1*]
delete **ctx**[*t2*]

XPath:

```
<xsl:template match="evt[ @ID = 'e1']/mod[ @lex = 'ready' ]" />
<xsl:template match="ctx[ @ID = 't2']" />
```

Figure 4.2.1-3

Transformational rules for condensing sentence (79)

Jing (2000) describes a sentence condensation (reduction) algorithm which operates over the syntactic parse tree and uses multiple sources of information to decide which portions of a sentence to elide. Although the transformation rules to remove a subtree/phrase are not explained, sample sentence reductions are reported. One

representative sentence (81) is shown below. Its corresponding reduction (82) (from human judges) is also shown.

(81) *When it arrives sometime next year in new TV sets, the V-chip will give parents a new and potentially revolutionary device to block out programs they don't want their children to see.*

(82) *The V-chip will give parents a device to block out programs they don't want their children to see.*

ctx [ID=*t1* TYPE=*sentence*
  rel[*when*
    ctx [ID=*t2*
      obj[ID=*o1* PRON=*it*]
      obj[ID=*o2* **mod**[*next*] **class**[*year*]]
      evt[ID=*e1* S=*o1* ACTION=*arrive* O=*o2* TENSE=*present*
        mod[*sometime*] **rel**[*in new TV sets*]]]]
  obj[ID=*o3* DET=*the* **name**[*V-chip*]]
  obj[ID=*o4* **class**[*parents*]]
  obj[ID=*o5* DET=*a* **mod**[*new and potentially revolutionary*] **class**[*device*]
    evt[ID=*e2* ACTION=*block* O=*o6* **mod**[*out*]]]
  obj[ID=*o6* **class**[*ptrograms*]]
    ctx [ID=*t3*
      obj[ID=*o7* PRON=*they*]
      obj[ID=*o8* **class**[*children*]]
      obj[ID=*o9* PRON=*they* **prop**[*o8*]]
      evt[ID=*e3* S=*o7* ACTION=*want* TENSE=*present* NEG
        evt[ID=*e4* S=*o8* ACTION=*see*]]]]]]
  evt[ID=*e5* S=*o3* ACTION=*give* MODAL=*will* O=*o5* IO=*o4*]]]

Figure 4.2.1-4

CAMEO representation for sentence (81)

Figure 4.2.1-4 lists the CAMEO representation and the operations to produce the condensed sentence are shown in Figure 4.2.1-5.

Pseudocode: delete **rel** containing **ctx**[*t2*]
delete **mod** contained in **obj**[*o5*]

XPath:

```
<xsl:template match="rel[ ctx[ @ID='t2' ] ]" />
<xsl:template match="obj[ @ID='o5' ]/mod" />
```

Figure 4.2.1-5

Transformational rules for condensing sentence (81)

As a final example of the transformational possibilities using the CAMEO representation language, I will use a sample sentence from Knight and Marcu (2000), who propose a noisy-channel statistical model of sentence condensation. In their model, an exhaustive list of syntactic tree transformations are encoded in a packed forest. From this, the trees are iteratively transformed and used to generate a large number of potential surface expressions. The surface expressions are ranked using a stochastic measure based on word-bigrams and a learned probabilistic model of a "noisy channel" for text expansion. Knight and Marcu show the example sentence (83) listed with the highest scoring reductions, according to their statistical measures.

The CAMEO representation for the sample sentence is given in Figure 4.2.1-6. Table 4.2.1-7 shows each of the sentence variations listed by Knight and Marcu, and the corresponding CAMEO operation which will produce it when applied to the representation. (The operations listed in the table are XPath expressions for use in template match attributes as shown previously).

(83) *Beyond that basic level, the operations of the three products vary widely.*

```
ctx [ ID=t1 TYPE=sentence
      rel[ beyond obj[ ID=o1 DET=that mod [ basic ] class[ level ]]]
      obj[ ID=o2 DET=the class[ operation ]
           rel[ of obj[ ID=o3 DET=the QUANT=three class[ products ]]]]
      evt[ ID=e1 S=o2 ACTION=vary TENSE=present mod[ widely ]]]
```

Figure 4.2.1-6

CAMEO representation for sentence (83)

Table 4.2.1-7
CAMEO transformations for variations of sentence (83)

Surface Text	CAMEO Transformation Rule
Beyond that level, the operations of the three products vary widely.	`obj[@ID='o1']/mod[@lex='basic']`
Beyond that basic level, the operations of the three products vary.	`evt[@ID='e1']/mod[@lex='widely]`
Beyond that level, the operations of the three products vary	`obj[@ID='o1']/mod[@lex='basic']` `evt[@ID='e1']/mod[@lex='widely']`
Beyond that basic level, the operations of the products vary	`obj[@ID='o1']/@QUANT` `evt[@ID='e1']/mod[@lex='widely']`
The operations of the three products vary widely	`ctx[@ID='t1']/rel`
The operations of the products vary widely	`ctx[@ID='t1']/rel` `obj[@ID='o3']/@QUANT`

The operations of the products vary	`ctx[@ID='t1']/rel`
	`obj[@ID='o3']/@QUANT`
	`evt[@ID='e1']/mod[@lex='widely']`

The operations of products vary	`ctx[@ID='t1']/rel`
	`obj[@ID='o3']/@QUANT`
	`obj[@ID='o3']/@DET`
	`evt[@ID='e1']/mod[@lex='widely']`

Operations of products vary	`ctx[@ID='t1']/rel`
	`obj[@ID='o2']/@DET`
	`obj[@ID='o3']/@QUANT`
	`obj[@ID='o3']/@DET`
	`evt[@ID='e1']/mod[@lex='widely']`

The operations vary	`ctx[@ID='t1']/rel`
	`obj[@ID='o2']/rel`
	`evt[@ID='e1']/mod[@lex='widely']`

Operations vary	`ctx[@ID='t1']/rel`
	`obj[@ID='o2']/@DET`
	`obj[@ID='o2']/rel`
	`evt[@ID='e1']/mod[@lex='widely']`

As the examples in this section demonstrate, CAMEO provides a simple and flexible interface for manipulation of the internal representation for tasks such as sentence condensation. Because syntactic relationships are explicitly encoded and objectified, patterns for matching constituents can be kept simple and precise. Transformational operations amount to deleting elements or attribute values, which can be accomplished by suppressing the item during an identity transformation. Once the representation has been transformed, the surface realiser generates the corresponding surface form in a deterministic operation. In the next section I will present this process in detail for a particular syntactic transformation, and compare the performance of CAMEO with an extant system.

4.2.2 Active/Passive alternation

Canning (2002) gives a treatment of the active/passive alternation of verb phrases in the SYSTAR text simplification system. She applies 14 rules to 33 instances of agentive passive verb phrases. The system achieves a 53% grammar and 57% meaning precision score based on the scores of four human judges. In this section I will present a comparative analysis of the output of the SYSTAR system and activisation using the CAMEO representation.

The SYSTAR system uses the output of the RASP processing tools and Canning does not attempt to hand-correct errors in the parse output. This is unfortunate as it conflates parser performance with the SYSTAR rules, making it difficult to evaluate the performance of the system. In my experiments, I used the same test sentences as reported in Canning (2002), treating them as unseen data with respect to the experimental framework. However, I corrected parse errors propagated to the CAMEO representation before attempting the manipulations. Therefore, comparison with the SYSTAR output is expository only, intended to aid analysis of the CAMEO output. Although Canning does not always note sentences with parse errors, where this apparently accounts for the discrepancy in output, I note in my analysis.

The test sentences presented by Canning are sorted into 7 syntactic categories. Two of the 33 sentences were unused due to a reported software bug. The unused sentences were not available and thus not included in my experiment. In addition to the main passive verb phrases reported by Canning, three of the sentences contain a secondary passive verb. Because CAMEO allows changing each verb phrase independently, examples are given for both activisations. Several of the sentences also included non-agentive passive forms (i.e. passive verb constructions with no expressed subject). These cannot be activised without deeper inferencing and are not addressed in the study.

I used the 31 sentences listed in Appendix A of Canning (2002) to test the activisation of sentences in CAMEO. I converted each sentence into the CAMEO representation using the procedure described in Section 3.7. The sentences were reviewed for parsing errors and manually corrected. The `PASSIVE` attribute was then

removed from each passive verb and the sentences transformed to surface text using the CAMEO surface realiser. The results were compared to those reported by Canning (ignoring minor punctuation discrepancies), and the complete output is listed in Appendix A. Table 4.2.2 summarizes the differences between the two systems.

Table 4.2.2
Comparison of CAMEO output to SYSTAR on test sentences

	Past	Present	Modal	Perfect	Progressive	Is to be	Total
Matches	10		1	3		1	15
Constituent order	4			1	1	1	7
SYSTAR Errors	7	2					9

The first row in Table 4.2.2 shows the number of sentences correctly activised by CAMEO that directly matched the SYSTAR system output for each syntactic category. The second row shows the number of sentences that differed only by constituent order. These sentences are correctly activised and judged grammatical. Without an independent judgment of the meaning of the sentence it is difficult to quantify the performance in this category, however, I will discuss several of the representative cases. The final row gives the number of sentences activised incorrectly by SYSTAR, due to system or parse errors. These sentences (with parsing errors corrected) were correctly activised by CAMEO.

One of the challenges for activising more complex passive constructions is constituent ordering. The position of adverbs and prepositional phrases in a sentence that has been activised is variable. For example consider sentence (84):

(84) *Emma Rae , 15 , of Parkhurst Road , said her three-year-old brother , James, suffered a broken leg when he was knocked down by a car* ***on Sunday , not far from where the little girl died*** *.*

The two final prepositional phrases (*on Sunday, not far from …*) are adverbial but take a post-object position in the passive sentence (i.e. after *a car*). When activising this sentence the passive object (*a car*) becomes the active subject. The two adverbial phrases can either follow the passive object, in which case they assume a post-subject position in the new sentence, or remain in a post-object position behind the new object (*he*). SYSTAR produces the former (85) and CAMEO the latter (86), shown below.

(85) *Emma Rae, 15, of Parkhurst Road said her three-year-old brother, James suffered a broken leg when a car* ***on Sunday***, ***not far from where the little girl died*** *knocked him down.*

(86) *Emma Rae , 15 , of Parkhurst road , said her three-year-old brother , James, suffered a broken leg when a car knocked him down* ***on Sunday***, ***not far from where the little girl died*** *.*

(The SYSTAR output was affected by a parse error, which incorrectly analysed parts of the sentence. However, the resulting activisation is not ungrammatical and illustrates the variability of the possible transformations.)

The prepositional phrases remain in the post-object position in CAMEO because the FORM attribute is set when the sentence is transformed into the CAMEO representation. The activisation only requires modifying the PASSIVE attribute, so the positions of the phrases do not move in relation to the *expressed* sentence structure. Of course, removing or modifying one or both of the FORM attributes is another option, which would allow producing the variation shown in (85), as well as many others. A rule-based system does not have this freedom because the rules are statically linked to the transformations. Thus a certain match will always produce the same output.

Another example where the constituent order significantly affects the readability of the sentence (87), is shown below with the SYSTAR (88) and CAMEO (89) outputs:

(87) *Last year the campaign was supported by 38 primary schools with a further five joining in this time .*

(88) *38 primary schools with a further five joining in this time supported last year the campaign.*

(89) *Last year 38 primary schools supported the campaign with a further five joining in this time .*

Because the matching rules in SYSTAR are relatively shallow, the constituents before the passive verb phrase are ignored, or effectively taken as the entire subject. As this example shows this is not always advantageous, and may produce more awkward

activisations. Because CAMEO encodes the syntactic dependencies in the representation, and the representation is used to transform text, the precise subject is used in the transformation. The resulting text is a closer match syntactically to the original passive sentence.

There was one tense discrepancy which I judged to be an error with the SYSTAR output. The sentence is reproduced below (90) with the output from SYSTAR (91) and CAMEO (92).

(90) *She is impressed by the changes in the city , particularly the proposed introduction of the Metro .*

(91) *The changes in the city impressed her particularly the proposed introduction of the Metro.*

(92) *The changes in the city impress her , particularly the proposed introduction of the Metro .*

The tense of the verb phrase is clearly *present* (i.e. *to be* conjugated *is*), where SYSTAR produces a corresponding past tense. However it is difficult to determine whether this is intrinsic in the system, since the only other present tensed example resulted in another SYSTAR error. The tense of the verb is analyzed and encoded in the CAMEO representation, ensuring that during surface realisation the tense is preserved and properly generated.

One of the difficulties in the activisation of sentences is the ambiguity that can sometimes appear in the passivised subject (see Section 3.4.6). Sentence (93) below, taken from the test set, is one example. The prepositional phrase [*by post*] is probably best interpreted as describing the *means* of the notification, rather than having the subject function. However, both systems activised the sentence as [*Post informed Mr. Clifford, a single man who is now on police bail.*]. To prevent such generalizations, a method for detecting exceptions to the general passive construction would be required, either during the transformation into the representation from the syntactic structure, or in the syntactic analyser itself.

(93) *Mr Clifford , a single man who is now on police bail , was informed by post .*

The remaining sentences which had errors in the SYSTAR output comprised constituent errors which changed the meaning of the sentence. For example, in sentence (94), the SYSTAR output incorrectly mixes the dependent clause with the passive verb phrase, resulting in an incorrect meaning (95). This may be due in part to a parser error, or it may be related to the shallow rule match expressions. The CAMEO output is shown in (96).

(94) *Alan , who is sponsored by Washington-based outdoor clothing and equipment manufacturer Berghaus , has now reached the summit of Makalu , the fifth highest peak in the world .*

(95) *Washington-based outdoor clothing and equipment manufacturer Berghaus has now reached the summit of Makalu, the fifth highest peak in the world sponsored Alan, who*

(96) *Alan , who washington-based outdoor clothing equipment manufacturer Berghaus sponsors , has now reached the summit of Makalu , the fifth highest peak in the world.*

Further instances of similar errors in the SYSTAR output are recorded in Appendix A. In each case, the corresponding CAMEO output is acceptable, and closely matches the constituent order of the original sentence.

In several cases the CAMEO output had minor punctuation errors relating to missing or misplaced commas. As mentioned in Section 4.1.3.2, the placement of commas during realisation is driven by embedded contexts. When a phrasal **ctx** element is encountered within a sentence-level **ctx** element, the surface realiser will offset the phrase with commas. This is a simplistic approach which is generally sufficient, but is inadequate in some cases. For example, the CAMEO output of sentence (4) in Appendix A is shown in (97) below. The representation includes embedded contexts for the phrase [*who treated the dead girl*] and [*the consultant who treated the dead girl*], resulting in an extra comma. The surface realisation algorithm could be enhanced to include a more sophisticated treatment of cases such as these, but a more general approach would be to add attributes to the representation that would allow higher-level

processors to control the expression of punctuation directly, or override the default behaviour of the surface realiser.

(97) *When demonstrators returned this morning , the consultant , who treated the dead girl , joined them .*

4.3 Conclusion

The structure and form of the CAMEO representation, along with the rich set of features for encoding surface information, has advantages over other representational forms for certain operations. In this chapter I examined two specific operations on the representation, surface realisation and sentence passivization, and compared CAMEO with other representational approaches on several examples from the literature.

The CAMEO language includes a strategy for deterministic surface realisation of text from the internal representation. The transformation algorithm uses an event-driven approach to process the recursive structure of the representation. The surface form is generated using the elements in the representation and their attribute values. Special attributes are defined to allow variations in the surface form.

The flexibility of the representation is reflected in its object-oriented design, which enables programmatic manipulation using existing DOM-based tools. Transformations of the internal representation were demonstrated for sentence condensation and sentence activisation tasks. The experiments showed that the representation has advantages over more linear and lexical approaches to these tasks, and has transformational simplicity on par with more deeply structural representations such as the f-structures of LFG.

5

Context in Symbolic Processing

In this chapter I will discuss the different kinds of contextual information available for use in text processing, and explain the representational approach I have taken. Through experimental analysis I will evaluate how certain types of discourse processing can leverage contextual information.

Research into the structure of discourse has shown that beyond the local level of adjacent sentences there emerges a wider scope of textual coherence, sometimes referred to as a discourse segment (Allen, 1995). In general, a discourse segment is a group of text which coheres to a certain topic. Each discourse segment produces a local context in which to interpret the text (in addition to other more global contexts). The local context contains the evolving state of the discourse segment and is critical to understanding the text. Without considering the local context, sentences would be interpreted in isolation and a discourse would not be possible.

There is currently no clear formal definition of what comprises a discourse segment or how they can be determined. This is an active area of research and includes the question of how sentences in a discourse segment internally relate to each other, and how discourse segments externally relate to other discourse segments. Hovy (1990) surveys many of the coherence relations proposed in the literature as holding between sentences. Rhetorical Structure Theory (Mann and Thompson, 1987), among others, gives an account of segmental relations. Much of this type of analysis is beyond the scope of this research. Instead, I will focus on what information can be derived from the surface syntactic analysis, in relation to discourse segments.

Although it is not clear what formally constitutes a discourse segment, several properties are apparent:

First, the sentences in a discourse segment necessarily share a common context and set of assumptions about the state of the discourse. This follows from the definition of a discourse segment, i.e. that the sentences in the segment cohere. In addition, a discourse segment should generally retain a single personal aspect throughout. For example, in a narrative the author sometimes assumes third person to relate the story. Within the story there may be segments from other sources, such as a speaker in a dialog. Each segment of dialog represents a new personal aspect, i.e. that of the speaker. Because they do not share a common reference with the global narrative, these dialogs should be considered as separate discourse segments.

Next, a discourse segment boundary represents some shift in focus or topic. Sometimes this is simply a shift in reference, as described above for dialog segments. Other times the shift may be more subtle and difficult to detect. However, when the topic or reference shifts, it affects the coherence of the text, and a new discourse segment is warranted.

Finally, discourse segments exhibit a recursive nature. By this I mean that discourse segments can be compositionally constructed of other discourse segments until an entire document is considered a single discourse segment. This is necessitated by the flexibility of natural language, which places few constraints on the structure of a discourse segment allowing them to exist inside other segments. Subsequently, discourse segments appear in many forms and configurations, (including parallel constructions in some cases). This flexibility becomes an important consideration in designing a representation.

5.1 Document Structure

Text that appears in documents usually is presented with a graphical structure. For example, chapters, sections and paragraphs are separated by whitespace, and quotations or references from other documents are often offset. Structural elements such as these are good candidates for discourse segment boundaries, as they usually entail some shift in topic or aspect. Like discourse segments, they can appear in recursive, compositional

structures. For example, chapters can be composed of sections containing sub-sections containing paragraphs.

Increasingly, this structure is explicitly marked in online documents using languages like XML, HTML, and DocBook (an XML schema for document publishing), making it much easier to recover during processing. Although this does not necessarily help the discovery of finer-grained discourse segments, it provides some level of segmentation that is somewhat deterministic.

5.2 Sentence Structure

It has been noted by researchers in early work on discourse segmentation (Polanyi and Scha, 1984), that the syntactic structure of sentences bears a resemblance to the structure of discourse segments in a document. That is, sentence phrasal structure is hierarchical and compositional, much like discourse segments. In fact, the dependency relations produced by some theories of discourse segmentation look similar to phrasal parse trees, and some discourse parsing frameworks use a single parser at both the discourse and sentence level (e.g. Forbes et al., 2001). This is a helpful insight because it suggests a common representation for structure at all levels of text.

For a discourse segment, the fundamental structure I am proposing is a container serving as a context for the sentences. Extending this paradigm to the sentence level seems plausible because the clausal construct can serve as the context for the individual component constituents of the clause. The different NP constituents share the same context (clause) and are related through a coherent topic (main verb).

This perspective on clausal segmentation appears to fit with other syntactic theories as well. For example, Allen (1995) discusses the idea of a local constituent domain, defined as the set of constituents subsumed in the nearest S or NP node (in a phrasal parse). This correlates with the idea of a clausal segment, where the segment represents the local domain dominated by an S node.

5.3 Context

A key function of the discourse segment is providing context for the elements it contains. Sometimes elements of this context are made explicit, as when a speaker is identified in a dialog statement. Other times the entire context must be inferred. When inducing the local context, the hierarchical structure of discourse segments can sometimes be of help.

For instance, a document ultimately has some source or author. The outermost context of the document should begin with this source as the contextual reference. As new contexts are discovered during processing, such as chapters, sections and paragraphs, this source would be inherited so that all text in the document shares this reference. However, if a paragraph includes some embedded context, such as a quotation, a new context with a different source would need to be created, overriding the default inheritance of the parent context.

The context for a discourse segment contains more than just the reference, of course. All of the discourse entities encountered in the text become part of the context, and the context progressively evolves based on the information contained in the text it includes. The context serves to record the state of all its member elements, and can be used in tasks such as information extraction, and reference resolution.

There are several levels of information which can be derived from contexts, some of which are beyond the scope of this research.

The semantic content of a context requires some level of semantic processing to discover. This has applications to tasks such as text understanding, where logical properties such as entailment are required. I will assume semantic processing to be handled by external processors and will not address it.

Syntactic information provided by the context includes discourse entities (i.e. **obj** elements recovered from the dependency structure), grammatical relations, and tense and aspect. These are not strictly a contextual phenomenon as they are derived from individual sentences. However, this information will be leveraged at the contextual levels to varying degrees in the experiments that follow.

Another level of information that a context provides is segmentation. Segmentation helps constrain complexity by giving boundaries to the scope of discourse properties. For example, during reference resolution, it may be helpful to consider not crossing certain contextual boundaries. This can reduce the number of antecedents to consider. (I will explore this hypothesis later in Section 5.9.)

5.4 Contexts in CAMEO

Because context is arguably the distinguishing feature of discourse segments, and because a similar concept can be applied at varying levels of textual organization, from the document level to the clausal level, I will adopt contexts as a unit of organization.

The CAMEO **ctx** element is a generalized abstraction of a contextual grouping. It is a representational element integrated with the other elements of the CAMEO representational language, giving CAMEO the capability to represent arbitrary recursive contexts. As with other aspects of the language, this allows for discourse segmentation to be represented in a theory-neutral manner. For example, discourse parsers which use discourse segments finer than those extracted from the graphical document structure can construct a hierarchy of **ctx** elements.

I have earlier described briefly the form of the **ctx** element in the CAMEO representation (see Section 3.5.2.1). Here I will explain how it is used to explicitly mark the contextual segmentation of a document at all levels. At the document level the context is recovered from the document structural mark-up (when available) or inferred using a document grammar. At the clausal level the context is derived from the syntactic parse information.

A clausal context is used wherever an S node (sentential phrase) is encountered in the parse tree. This includes conjunctive, dependent and relative clauses. The context becomes a container for the other elements dominated by the S node (**obj**, **evt**, etc.). In addition, the top level clausal context of a sentence is placed inside a sentential context. (This facilitates reference by sentence number in both processing and generation.)

Document level contexts are used to represent structural mark-up during initial processing. Several specific mark-ups are supported directly in the framework (e.g. title,

paragraph, and author). Other types of mark-up are generalized using a contextual node with the original tag name as an attribute (TYPE). This allows various formats to be processed while still retaining the structural segmentation.

The TYPE attribute identifies the type of context (i.e. chapter, paragraph, clause, etc.). Other attributes are optional and depend on the context type. For instance, document, chapter, and section contexts may have a TITLE attribute. Clausal contexts optionally have a CONJ attribute as described in Section 3.4.2. A REF attribute can be attached to any context and is used to determine the author or source.

During processing, contexts are processed recursively. Each new context encountered becomes the new active context for the purposes of processing. The elements contained within the context constitute the local domain. For instance, a paragraph context will have sentence elements that constitute its local domain. A clausal context will have **obj**, **evt**, **mod**, and **rel** elements for its local domain. As processing completes on each context, the active context reverts to the parent and processing continues until all contexts have been processed. This approach effectively implements a contextual stack, which can properly handle the hierarchical nature of contexts.

The recursive property of **ctx** elements gives a complete representation for the document structure through the clausal level. In addition to its utility in processing tasks, this representation can be used to extend the surface generation algorithm of Chapter 4 to include document structure (e.g. paragraph breaks, chapter titles, etc.). Having document structure integrated with syntactic structure in a single representational form makes it unnecessary to have multiple versions of the document for different applications.

Before exploring some of the contextual issues relating to coreference that arise in text processing (which require a representation of context), I will first introduce the general problem of coreference resolution, survey some of the current approaches, and explain how the CAMEO representation helps facilitate this type of processing.

5.5 Reference Resolution

One of the critical tasks in the processing of discourse text is the resolution of anaphora and coreferences. Applications that require any more than a shallow semantic analysis will benefit from reference resolution, due to the high frequency of this phenomenon in natural language. Without a means for resolving coreferences, objects in a representational model remain as separate individuals and text understanding beyond the phrasal level is not possible.

Although anaphora is a general term describing references to entities in a discourse, it is sometimes taken to mean pronominal references in the reference resolution literature. The next few subsections will briefly survey the existing approaches to pronominal anaphora resolution and the more general problem of coreference resolution.

5.5.1 Algorithms for Pronominal Anaphora Resolution

Anaphora resolution is difficult because it is often ambiguous, and may need commonsense knowledge to resolve in some cases. For example,

(98) *John took his dog Fido to the vet. He drove very fast.*

A naïve resolution algorithm might pick Fido as the referent of *he*, based on recency or other textual features. To understand why this is a difficult problem, consider the following sentences which have an identical syntactic construction:

(99) *John took his friend Bill to the doctor. He was very sick.*

The contrast between these two cases suggests that an algorithm that does not take into account semantic context will not be able to resolve all anaphora correctly.

In addition to semantics, another important factor in anaphora resolution is a discourse model. A discourse model accounts for the attentional focus of a document, providing a basis for selecting one referent over another. For example, centering theory proposed by Grosz, Joshi, and Weinstein (1995), includes a discourse model that can be used in anaphora resolution algorithms. The claim made in centering theory is that an

attentional "center" exists at any point in a discourse, and should receive preferential treatment during anaphora resolution. Centering theory has been extended and has formed the basis of several resolution algorithms in the literature (e.g. Walker, 1998).

In contrast to the discourse model of centering theory, a purely syntactic approach was proposed by Hobbs (1977). Hobbs describes a "tree-walking" algorithm, which uses the structure of the syntactic parse tree to find antecedents. In this approach, a set of rules determines how the parse tree is traversed and ultimately settles on a referent. Although the tree searching of Hobbs is syntactically based, it does implicitly encode salience for grammatical roles in the rules of the algorithm (i.e. trees are traversed breadth-first, from left to right, giving preference to subject roles).

Salience factors are the central aspect to the anaphora resolution algorithm proposed by Lappin and Leass (1994), whose use of weighted features lends itself naturally to a computational model. The algorithm tabulates a set of salience features for referents in the text, updating them as each sentence is processed. When an anaphoric reference is encountered, the table is consulted to determine the most likely referent, based on its score in the table. The salience features used in the Lappin and Leass algorithm include locally derived properties such as syntactic constituency, and a few global properties such as recency. Semantic properties were not considered in the original algorithm.

The Lappin and Leass algorithm has received some attention in the research community partly because it fits well with computational techniques and has proven a fair approximation of anaphoric phenomena in natural language. However, one disadvantage of the model is the weights on the various salience features need to be determined experimentally. There is also some question about whether these weights are domain dependent.

5.5.2 Algorithms for Coreference Resolution

Coreference resolution expands the task of pronominal anaphora resolution to all referential phrases in the text. This includes not only all nominals (objects) but verbs (events) as well (although few studies of coreference resolution include verbs). Examples of work that acknowledge the more general problem of coreference are Alshawi and Crouch (1992), and Popescu-Belis and Robba (1997). Alshawi and Crouch

propose coreference resolution as a unification of nominal objects under QLF. Popescu-Belis and Robba simply propose a framework for representing referring expressions (REs) that can be extended to support experimental algorithms.

Recently there has been some effort at applying machine learning to reference resolution (and the more narrow problem of pronominal anaphora resolution) (Aone and Bennet, 1996; McCarthy and Lehnert, 1995; Ge et al., 1998; Soon et al., 2001). Machine learning algorithms cast the reference resolution task as essentially a classification problem, where a pair of references are labelled as coreferring (or not) based on a set of features extracted from the text (Ng and Cardie, 2001). Preiss (2002) compares the performance of a machine learning anaphora resolution algorithm to the shallow parse approach of Kennedy and Boguraev (1996) and finds no significant difference in performance.

Algorithms for full reference resolution must arrange the nouns that appear in a corpus into a set of equivalence classes. Each instance of a noun, whether anaphoric or not, refers to some particular conceptual entity of the author. All references in a discourse referring to the same entity form an equivalence class for that entity. Determining these equivalence classes is the goal of coreference resolution algorithms (Hirschman, 1997).

5.6 Reference Resolution in CAMEO

Implementation of a coreference resolution algorithm using CAMEO is facilitated by the representation of objects in the CAMEO language. Because objects are represented explicitly, and attributes can easily be attached to them, a resolution algorithm is essentially a unification operation over existing objects. Section 3.2.2 gives the object unification function defined by the representation which is used to test basic compatibility between objects. Reference resolution algorithms can use this to filter reference candidates before applying algorithm specific decision processing. Further, the contexts included in the representation allow for extending existing algorithms to factor in contextual features as well. For example, a by-line may have location information that is more likely to corefer with demonstrative pronouns.

As an example of the kinds of operations that can be implemented for coreference resolution, consider the two sentences and their corresponding representations:

John has a new car. He bought it yesterday.

ctx
[TYPE=sentence
obj[ID=o1 A=h G=m **name**[*John*]]
obj[ID=o2 DET=a A=i **mod**[*new*] **class**[*car*]]
evt[S=o1 ACTION=have O=o2 TENSE=present]
]
ctx
[TYPE=sentence
obj[ID=o3 PRON=he]
obj[ID=o4 PRON=it]
evt[S=o3 ACTION=buy O=o4 TENSE=past **mod**[*yesterday*]]
]

In this case there are two pronominal references to resolve: **id**(*o3*) and **id**(*o4*). Assuming the attributes shown in this example (i.e. animacy A and gender G) have been populated by some processing module, a simple resolution operation might proceed as follows. Once the pronoun **id**(*o3*) is encountered, the algorithm searches for the most recent object having an animacy attribute equal to *h* (human), and gender attribute equal to *m* (male). It can do this by explicitly recursing through all previous **obj** elements, or alternatively using a simple query of the form

Find the most recent **obj** *outside this context with* A=[*h*] *and* G=[*m*]. *Return the object.*

Of course this example is trivial and a more sophisticated algorithm will require determining many other properties of an object. For example, to resolve the pronoun **id**(*o4*), an algorithm may need to determine if a candidate object (say **id**(*o2*)) participates in the same verb. A simple test of the form

Find an **evt** *with* (S=[*o4*] *or* O=[*o4*] *or* IO=[*o4*]) *and* (S=[*o2*] *or* O=[*o2*] *or* IO=[*o2*]).

This query returns an event if both **id**(*o4*) and **id**(*o2*) are arguments to the same verb (i.e. subject, object, or indirect object). The possibilities for creating queries in CAMEO are as flexible as the XPath language which is used for the implementation. An example of a real query in the XPath language used in the experiments described below is

```
id('o4')
[ .//@idref='o2' or
  .//@obj  ='o2' or
  prop
  [ @obj = 'o2' or
    id(@obj)//@idref='o2' or
    id(@obj)//@obj  ='o2'
  ]
].
```

This query tests if **id**(*o2*) is in the NP domain of **id**(*o4*) by testing for any reference to **id**(*o2*) that can be reached through an element contained in **id**(*o4*), or contained in an object property of **id**(*o4*).

As demonstrated here, complex queries regarding elements, attributes and their relations can be accommodated by the representation. Explicit programmatic processing of the elements in the DOM is also possible, as well as a combination of both. The results of the experiment that follows were obtained using both types of operations on the representation provided by CAMEO. No other task-specific transformations were required.

5.7 Contextual Issues with Reference Resolution

Much of the work on pronominal reference resolution over the last decade has focused on syntactic and morphological processing. Performance achieved using these methods generally fall below 80% (Mitkov, 2001; Tetreault, 2001). To achieve incremental gains in performance of these systems, researchers have been exploring the application of deeper informational content to existing algorithms, such as semantic and pragmatic analysis. One aspect of language processing which has recently received some attention is the consideration of structure for reference resolution.

There are two types of structure to consider for reference resolution. One is the structure of the discourse, and one is the structure of the document. Discourse structure is a deep analysis of the semantic and pragmatic implications of sentences or utterances. For example, Tetreault and Allen (2004) experiment with several discourse segmentation strategies applied to dialog reference resolution.Their findings suggest that discourse structure does not provide a significant increase in performance, though a small incremental gain was reported. Other theoretical work, such as Ide and Cristea (2000) also suggests that leveraging discourse structure for reference resolution is

beneficial for a certain small percentage of constructions. This type of structural information is difficult to produce automatically, and is not generally available in annotated corpora.

The second type of structural information that is relevant to reference resolution is the textual/logical structure of a document. As mentioned above this is much more accessible for computational processing. Goecke and Witt (2006) present a corpus study on integration of document structure with anaphora resolution. They suggest that the hierarchical arrangement of document structure might influence an algorithm's choice of antecedent in two ways: either through the location of the referent in the structure, or through the effect of the structure on the search window.

In the first case, the authors suggest that an antecedent is more likely to occur in a structural element at the same level or higher in the hierarchy as the referent. For example, a referent in a document's section, say, 4.3 is more likely to have an antecedent in section 3.2, than sections 4.2.1, even though an antecedent in section 4.2.1 is likely nearer to the referent.

The second case suggested by Goecke and Witt (2006) is the effect the hierarchical structure of a document might have on a reference resolution algorithm's antecedent search window. For many implementations of reference resolution, a heuristic limit to the search space is employed. This may be 2 or 3 sentences, or it may be based on the number of candidate antecedents encountered. However, this does not take into account the hierarchical structure of the document. For example, a large list or quotation may intercede between an antecedent and a referent. Measuring the linear (i.e. non-hierarchical) distance may show the antecedent out of the algorithm's search window. Using the hierarchial structure, the intervening list or quote context is 'collapsed' and does not extend the size of the search window.

A related application of the hierarchical document structure is the consideration of contextual segment boundaries. In their corpus study, Goecke and Witt (2006) find that referents occurring in the middle through the end of a paragraph tend to have antecedents within the same paragraph. Referents that occur at the beginning of a paragraph tend to have antecedents within a much larger scope outside of the referent's paragraph.

In the following sub-sections, I will explore several challenges to reference resolution for specific types of anaphora and specific genres of text, based on a small corpus study. These issues serve to illustrate some of the specialized contexts that have to be directly addressed in order to move reference resolution performance past the current plateau achieved through generalized algorithms. An integrated representation of document structural context enables existing reference resolution algorithms to be extended to encompass contextual information. In Section 5.8 I will present several experiments which integrate contextual information from the CAMEO representation with a reference resolution algoritm.

5.7.1 Demonstratives in Context

One challenge faced when analyzing certain textual documents for coreference resolution is demonstrative pronouns (i.e. *this* and *that*). Many times a reference will be made to a particular portion of a document, or the document itself. For example, [*This document describes how to install ...*], or [*This section will explain ...*].

An even more difficult construction is found when the demonstrative is alluding in some way to the section title. For example, the following structure appears in a Linux HOWTO document:

```
2. Comparing Linux with other Operating Systems
2.1. General Comparison_GC
The best place to find out about this_GC is in such documents as
the `Linux Info sheet' , `Linux Meta FAQ ' and `Linux FAQ ' (see
"Linux Documentation") .
```

Notice that the demonstrative [this$_{GC}$] is referring obliquely in some way to the act of performing a general comparison of Linux with other operating systems, although in most cases resolving this to the section title would be adequate.

In each of these cases, some notion of context along with some concrete representation is needed to correctly resolve the reference. For the demonstrative [*this section*], simply resolving all like instances into an equivalency class would not be correct, as the reference is contextually dependent. Also, for tasks involving semantic processing, the root antecedent should link to the relevant portion of the document.

The contextual representation proposed in this section addresses both of these issues. The demonstrative can be evaluated with respect to the context (section) it is found in, and every context can be used as an antecedent, just like **obj** and **evt** elements.

5.7.2 Contextual Issues with First and Second Person

There is little mention of first and second person resolution in anaphora resolution research. It is generally assumed that these pronouns are unambiguous and should therefore form an equivalence class. However, in many cases such as news text and literature, contextual issues arise which make this assumption incorrect. Consider the following text which appears in the corpus of Siddharthan (2003). (Note: co-referents are labelled using the primary antecedent for clarity).

The Wolf$_{WOLF}$ and the Lamb$_{LAMB}$.

Once upon a time a Wolf$_{WOLF}$ was lapping at a spring on a hillside, when, looking up, what should he$_{WOLF}$ see but a Lamb$_{LAMB}$ just beginning to drink a little lower down. "There's my$_{WOLF}$ supper," thought he$_{WOLF}$, "if only I$_{WOLF}$ can find some excuse to seize it." Then he$_{WOLF}$ called out to the Lamb$_{LAMB}$, "How dare you$_{LAMB}$ muddle the water from which I$_{WOLF}$ am drinking?"

"Nay, master$_{WOLF}$, nay," said Lambikin$_{LAMB}$; "if the water be muddy up there, I$_{LAMB}$ cannot be the cause of it, for it runs down from you$_{WOLF}$ to me$_{LAMB}$."

"Well, then," said the Wolf$_{WOLF}$, "why did you$_{LAMB}$ call me$_{WOLF}$ bad names this time last year?"

"That cannot be," said the Lamb$_{LAMB}$; "I$_{LAMB}$ am only six months old."

"I$_{WOLF}$ don't care," snarled the Wolf$_{WOLF}$; "if it was not you$_{LAMB}$ it was your$_{LAMB}$ father;" and with that he$_{WOLF}$ rushed upon the poor little Lamb$_{LAMB}$ and ate her$_{LAMB}$ all up. But before she$_{LAMB}$ died she$_{LAMB}$ gasped out "Any excuse will serve a tyrant."

In this short text, there are seven instances of a first person pronoun which refer to two different individuals – neither of which is the narrator. Likewise, there are five instances of a second person pronoun, alternating between referents. To correctly process the text requires a representation of the context of each clause.

The CAMEO representation of the contextual structure (omitting certain clausal contexts) for the first two sentences is approximated below.

ctx [TYPE=*doc* TITLE= *The Wolf and the Lamb*
  ctx [TYPE=*para*
    ctx [TYPE=*clause* [*Once upon a time ...*]
    ctx [TYPE=*clause*
      obj [ID=*o11* [*he*]]
      ctx [ID=*t5* TYPE=*clause* REF=*o11* [*"There's my supper"*]]
      ctx [TYPE=*clause* REF=*o11* [*"if only I can find some excuse to seize it"*]]
      evt [TYPE=*clause* S=*o11* C=*t5* [*thought he*]]

The dialog produces context elements which are distinguished from the running text using the REF attribute. The REF attribute records the speaker and allows for the first (and possibly second) person pronouns to be correctly attributed.

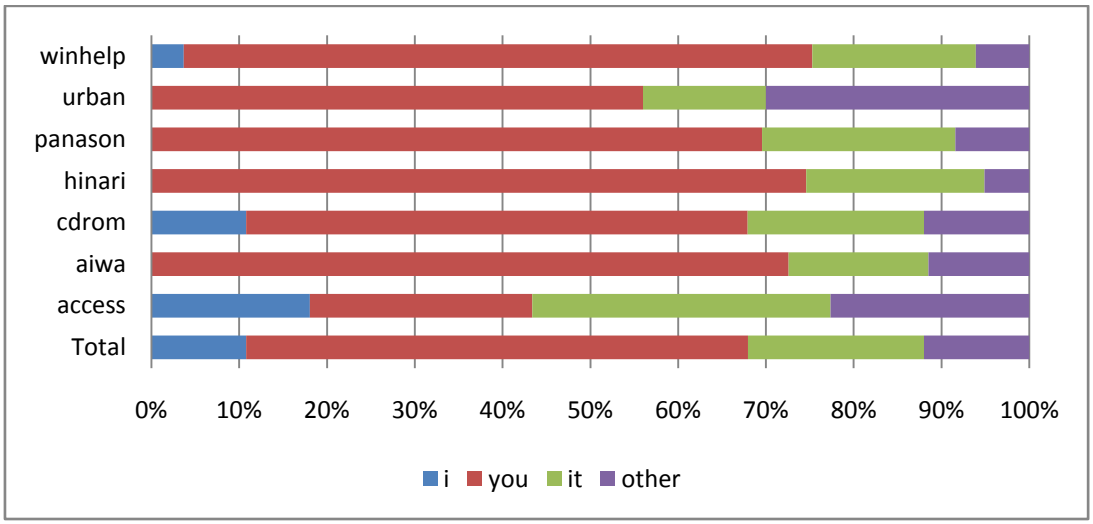
Note also that the pronoun [*he*] in the first sentence of the last paragraph can be resolved correctly because the dialog is out of the linear ordering of the text contexts. A syntactic, recency based reference resolution algorithm would choose [WOLF] as the antecedent because it is the most recent linear context. Without a contextual representation, the most recent linear context would instead incorrectly choose [FATHER].

Of course, not every discourse contains dialog, but there are other constructions where contexts are crucial to interpreting first and second person. These appear with varying degrees of frequency based on the genre of text and the intended purpose of the document.

Figure 5.7.2 gives the percentage distribution of pronouns for a sample corpus. The corpus is from the University of Wolverhampton (Mitkov et al., 2000) and is comprised of seven technical documents. The corpus has coreferential nominal identity chains marked in accordance with the MUC-7 syntax. The documents consist of:

Winhelp	README file for Windows Internet Explorer
Urban	DIY article about tools and tool safety
Panason	Instruction manual for a Panasonic television
Hinari	Instruction manual for a Hinari television
Cdrom	Linux CD-ROM HOWTO
Aiwa	Instruction manual for Aiwa stereo receiver
Access	Linux adaptive technology HOWTO

Figure 5.7.2 – Percentage distributions of several pronouns on the Wolverhampton corpus



The graph reveals that the most common pronoun in this corpus (on average) is [*you*], which comprises nearly 60%. Since the corpus is comprised of generally instructive documents this is to be expected. The next most common pronoun is [*it*], followed by [*I*] and finally, all others combined (this includes *he*, *she*, *they*, *one*, *everybody*, etc.)

Looking at the distribution of the pronoun [*I*], it is clear even within this specific genre of text, that there is high variance. Only three of the seven documents use first person, and the usage varies. The *access* and *cdrom* documents are both written in first person and aimed towards users of the Linux operating system. In addition, the *cdrom* document also employs a rhetorical first person in questions used as section titles (e.g. *How can I tell what speed CD-ROM I have ?*). The *winhelp* document on the other hand, quotes questions from users in the first person.

The de facto algorithm for resolving first person references, based on number and person constraints, simply links to the most recent preceding instance. Using this algorithm on the *access* document results in a 90.9% precision for first person pronouns. (For these experiments precision was measured as membership in the correct equivalence class).

An enhanced algorithm which takes into account contexts can be implemented using a back-off strategy: The contextual chain is searched for the nearest ancestral context which has a reference attribute. If no context is found, the system falls back to the default strategy of resolving to the nearest preceding instance of the pronoun. This strategy correctly resolves 100% of the first person pronouns in the *access* document when contextual information is included in the representation.

The same strategy can be employed when resolving second person pronouns, although this can be much more difficult. If the pronoun occurs outside of any special context, it can be resolved to a single equivalence class (representing the reader). Otherwise, some means of determining the second person focus of the context is necessary. (In some cases this may also be the equivalent to the reader.)

For certain classes of text, respecting context for first and second pronouns can improve the performance of reference resolution. The gain will vary widely depending on the distribution and syntactic constructions of the document. However, when text understanding is the goal, an analysis which includes context will be crucial.

5.7.3 Third Person Anaphora in Context

Although third person anaphora is generally considered a local phenomenon, it poses certain difficulties both with and without contextual considerations. In a previous section I described how a third person pronoun can bind to a first person pronoun in the context of dialog. However, in general, a dialog (or other embedded) context contains a separate table of salient referents, independent of the main context. For instance, in this passage from Lewis Carroll's *Alice In Wonderland*, the third person pronouns do not corefer inside and outside the dialog, although they intervene:

Presently she$_{ALICE}$ began again. "I wonder if I shall fall right THROUGH the earth! How funny it'll seem to come out among the people that walk with their heads downward! The Antipathies, I think-- " she$_{ALICE}$ was rather glad there WAS no one listening, this time, as it$_{ANTIPATHIES}$ didn't sound at all the right word) "--but I shall have to ask them what the name$_{NAME}$ of the country is, you know. Please, Ma'am$_{MA'AM}$, is this New Zealand or Australia?" (and she$_{ALICE}$ tried to curtsey as she$_{ALICE}$ spoke--fancy CURTSEYING as you're falling through the air! Do you think you could manage it$_{CURTSEYING}$?) "And what an ignorant little girl she$_{MA'AM}$'ll think me for asking! No, it'll never do to ask: perhaps I shall see it$_{NAME}$ written up somewhere."

The parenthetical aside inserted into the dialog introduces a new third person referent interposed between the final [*she*] and the antecedent [*Ma'am*]. This construction can only be resolved by considering these segments as separate contexts. A similar construction exists for [*it*$_{NAME}$], where a local anaphoric reference to [*it*$_{CURTSEYING}$] intervenes between the final [*it*$_{NAME}$] and its antecedent.

Notice however, that the first anaphoric [*it*$_{ANTIPATHIES}$], which occurs in the narrative context, refers to an antecedent *inside* a dialog context. So even though an embedded context is generally associated with an independent table of referents (i.e. discourse entities), these are accessible to the parent context.

Unfortunately, the opposite situation can also occur. The following is an example taken from the Siddharthan (2003) corpus of a reference [*it*$_{BONE}$] that occurs inside a dialog context to an antecedent that exists in the narrative context.

A Wolf had been gorging on an animal he had killed, when suddenly a small bone in the meat stuck in his throat and he could not swallow it. He soon felt terrible pain in his throat, and ran up and down groaning and groaning and seeking for something to relieve the pain. He tried to induce every one he met to remove the bone$_{BONE}$. "I would give anything," said he, "if you would take it$_{BONE}$ out."

These types of constructions further complicate resolution as antecedents can appear in both hierarchical proximity (ignoring intervening contexts) and linear proximity.

One other notable property of third person anaphora which must be considered is the ability to block potential antecedents. Consider the following sentence which is a continuation of the previous passage:

Then the Crane put its long neck down the Wolf's throat, and with its beak loosened the bone, till at last it_{CRANE} got it_{BONE} out.

The final [it_{BONE}] is blocked from binding to the preceding [it_{CRANE}] because of syntactic constraints (i.e. both pronouns are members of the same local domain). Most reference resolution algorithms will code for these constraints and would not have a problem resolving this type of construction. However it is easy to change the sentence so syntactic constraints on the two pronouns are not violated.

Then the Crane put its long neck down the Wolf's throat, and with its beak loosened the bone, till at last it_{CRANE} believed it_{BONE} was out.

The revised sentence circumvents the blocking effects and requires some other means to determine the binding of the second pronoun, in this case (and many others), either a semantic analysis is required, or some more sophisticated constraint processing must be used.

5.8 Evaluating Resolution

Evaluation metrics for reference resolution often utilize standard calculations for precision and recall. However, because a reference resolution system relies on many different processing components, it is sometimes difficult to measure the success of the algorithm, or the effects of different theories. Mitkov (2002) proposes several different metrics designed to give a more uniform and precise evaluation of different reference resolution systems. He argues that recall is not relevant to robust resolution systems because they typically give values for all anaphora in a text. Additionally, variance in how recall is calculated makes direct comparison difficult. For example, some researchers may use the total number of anaphora (from the annotation key), while others may use the total number identified by the system.

Instead of the standard *F*-score based on precision and recall, Mitkov (2002) suggests *success rate* which is a simple ratio of the correctly resolved anaphora to all

anaphora in the text. Additionally, he proposes several other measures designed to evaluate specific classes of anaphora, which he refers to as *critical* and *non-critical*. Finally, he argues that to evaluate a reference resolution algorithm, versus a reference resolution system, the exact *success rate* can be found by hand-correcting the output of any pre-processing stages to ensure that the input to the algorithm is correct.

The standard calculation of *F*-score, and the alternate scoring scheme proposed by Mitkov (2002), assume a single-stage system which produces a single antecedent that can be judged on a binary scale (i.e. right or wrong). This imposes an unnecessary limitation on the application of a reference resolution system because it does not consider the possibility for multi-stage processing.

In the preceding sections I discussed some of the challenges faced when attempting anaphora resolution. Performance of existing algorithms using syntactic and lexical constraints is generally reported to be in the 60-80% range. Further gains usually require analysis beyond the surface level, addressing lexical and semantic constraints. (See Section 6.3). Because of this, evaluating a pronoun resolution algorithm strictly on a binary precision metric does not measure its true potential. Basic algorithms can be employed to do initial filtering and pass the results to more refined methods. A simple binary metric ignores this possibility and only scores a single outcome.

For example, many resolution algorithms begin with a list of possible antecedents and apply a series of scoring constraints before selecting the highest scoring antecedent. If there are many cases which require deep semantic interpretation, it may be that an algorithm performs poorly because it happens to select (usually through some heuristic) the wrong antecedent from a list of two or more having similar scores. However, if it could be shown that the algorithm returns the true antecedent in a small set of its highest ranking antecedents, the algorithm should improve the overall performance when combined with subsequent deeper processing stages (e.g. a semantic analyzer).

In the experiments that follow, I propose using evaluation criteria that measures the precision and accuracy based on a *ranked list* of likely antecedents, rather than a single antecedent. The antecedents appear on the list ranked by salience, which for a naïve algorithm can simply be proximity to the referent. Using a ranked list in the evaluation metric calculations gives a better indication of how the algorithm would perform in a

system that includes a second stage processor. The goal of the first stage algorithm is to produce the smallest set of candidates which contains the correct antecedent.

The new evaluation metric I am proposing is derived in the following manner. The standard definitions of precision *P*, recall *R*, and *F-score* are calculated normally except the *ranked list* of likely antecedents is used when determining correctness. That is, an anaphor is determined to be resolved correctly if any antecedent in a proposed set is correct. This produces exactly the score that would be achieved given a hypothetical, infallible second stage processor to select from the proposed set of antecedents.

This revised *F-score* alone would be unhelpful, as a naïve algorithm which simply included all preceding nouns would score perfectly. Thus I propose a new measure called the antecedent focus. The antecedent focus AF of a resolved anaphor is calculated by determining the rank of the true antecedent in the proposed set of antecedents, together with the number of antecedents in the set.

$$AF = \frac{1}{R + |S|}$$

Where R is the zero-relative rank of the true antecedent in the set S, and $|S|$ is the cardinality of S. When the true antecedent is found in a set containing exactly one antecedent, this reduces to

$$AF = \frac{1}{0 + 1} = 1$$

When the true antecedent is not contained in S, R is set to infinity yielding $AF = 0$. The antecedent focus balances the rank of the true antecedent with the size of the proposed set. Both the rank and set size need to be small to achieve a high antecedent focus.

The average *AF* over all anaphora gives an indication of how well the algorithm performs without a second stage processor. As an example, an algorithm that always returned 2 antecedents, with the true antecedent first, would score $AF = 0.5$. The same algorithm returning the true antecedent second would score $AF = 0.33$.

The revised *F-score* and *AF* score together give a better indication of the true performance of the algorithm compared with an *F-score* based on a binary precision. In the experiments that follow, I will report the revised *F-score* along with the *AF* score in my analysis.

5.9 Experiments

For these experiments, I implemented the syntactic approach of Hobbs (1978) as summarized in Jurafsky and Martin (2000). The basic algorithm consists of retracing the parse tree from the anaphor in a constrained way to find the most recent plausible antecedent. If the sentence containing the anaphor does not yield an antecedent, preceding sentences are explored in order of most recent first. Each proposed antecedent is tested against the basic constraints of number, gender, and animacy.

The Hobbs algorithm, though it uses a somewhat difference approach, achieves scores comparable with the Lappin and Leass (1994) salience table approach. Many of the salience weights which are explicitly encoded in the Lappin and Leass algorithm are implicitly accounted for by Hobbs. I chose to use the Hobbs algorithm because its recursive nature fits naturally with the contextual model at hand. (The implementation of the algorithm was written in a single XSLT transformation run over the CAMEO XML file.) Also, the fact that the clausal contexts in the CAMEO representation equate to syntactic local domains simplifies the processing of some steps in the Hobbs algorithm.

There is no provision in either Hobbs or Lappin and Leass for determining a set of antecedents, as I proposed for the performance metric. Both algorithms produce the best candidate based on recency and other constraints. Presumably the second best candidate would then be the next ranked antecedent, or the next most recent. Because there is no way to determine the optimum set size, I ran several tests using different set limits. When contextual information is considered, the algorithm can be stopped when crossing certain contextual boundaries. This allows a comparison of the *AF* based on the segmentation provided by the contextual information, versus a longer list of antecedents.

For example, if the set size was limited to 6 antecedents, and the contextual segmentation produced an average set size of 3 antecedents with little change in precision or recall, the segmentation is improving the *AF* of the algorithm.

5.9.1 Testing Embedded Contexts

As I discussed earlier in the chapter, the genre and style of text will determine the extent that contextual processing can affect the performance of text processing. Documents that contain no dialog or other embedded textual constructs will show little if any improvement on tasks such as reference resolution when evaluated against a strict precision metric. Documents such as literature and certain types of news stories, will benefit from the application of contextual processing, to the extent that the text contains contextual structure.

To test this hypothesis, I used a document (`literature.txt`) from the Siddharthan (2003) corpus which contains a fair amount of dialog. The document contains several short stories and several excerpts from longer works.

Basic structural contexts were added to the document for section and paragraph breaks, and a custom tokenizer was used to automatically detect and insert contexts for dialog. Each dialog initiated a new context container and inside of this container sentences were processed normally. The document was then parsed using the RASP system, and the output was converted into the CAMEO representation as described in Chapter 3.

The Hobbs reference resolution algorithm was run over the CAMEO representation for third person pronouns only, with and without contextual processing. With contextual processing, the algorithm did not treat sentences neighbouring a dialog context as adjacent to the sentences inside the dialog context. Without contextual processing, these sentences were treated as adjacent and processed normally.

The results are shown in Table 5.9.1 for several different set sizes. The table gives the *F-score*, Precision (*P*), Recall (*R*), and Antecedent Focus (*AF*) for each size with

and without contexts. Note when the set size is set to 1, the *AF* metric produces the same result as standard (binary) precision.

For this corpus, the contextual information improves the precision and the *AF*, especially for smaller set sizes. As the set size is relaxed, more antecedents are allowed and the precision for both algorithms improve and approach parity. Additionally, the *AF* degrades as more antecedents dilute the precision, until the two algorithms are at near parity.

Table 5.9.1– F-score, Precision, Recall, and AF for third person resolution using several set sizes

	Contexts				No Contexts			
$\|S\|$	F	P	R	AF	F	P	R	AF
1	48.3	51.4	45.5	51.4	43.4	44.7	42.1	44.7
2	70.3	74.9	66.3	45.3	66.3	68.4	64.4	40.8
3	73.4	76.8	70.3	33.2	73.0	75.3	70.8	32.4
4	78.1	81.6	74.8	27.7	77.5	80.0	75.2	26.8

The contextual information is shown here to give an advantage for this text. When a unary antecedent set size is evaluated, precision, recall and *AF* were higher with the contextual information. In fact, precision and *AF* were higher for all set sizes tested, though the difference becomes diminished as the size increases.

5.9.2 Contextual Segmentation

Texts which do not include embedded contexts but do include some structure, may benefit to a lesser degree by using the explicit contextual segmentation found in the document structure to limit complexity during processing, much like the embedded contexts did in the previous section.

To test this hypothesis I used the Hobbs coreference algorithm on an annotated corpus to propose a set of antecedents for third person pronouns. Because the algorithm does not include a determination of the optimal set size, I tested a range of set sizes,

from 1 to 10. Generally, resolution algorithms are run with a limit on sentence recency. For example, a common heuristic is to limit the algorithm to 2-4 sentence histories. However, the number of possible antecedents varies with the content of a sentence, so different groups of 2-4 sentences can potentially have a wide range of possible antecedents. In these experiments the antecedent set size is the limiting factor, regardless of the sentence distances.

The corpus used in the experiments was comprised of seven technical documents from the University of Wolverhampton (Mitkov, 2000). The corpus has coreferential nominal identity chains marked in accordance with the MUC-7 syntax. Because the document structure was not explicitly annotated, I added basic structural information based on textual elements which included sections, paragraphs, titles, etc. The documents were then processed using the RASP system, and the output was converted to the CAMEO representation. The results are presented in Figure 5.9.2.

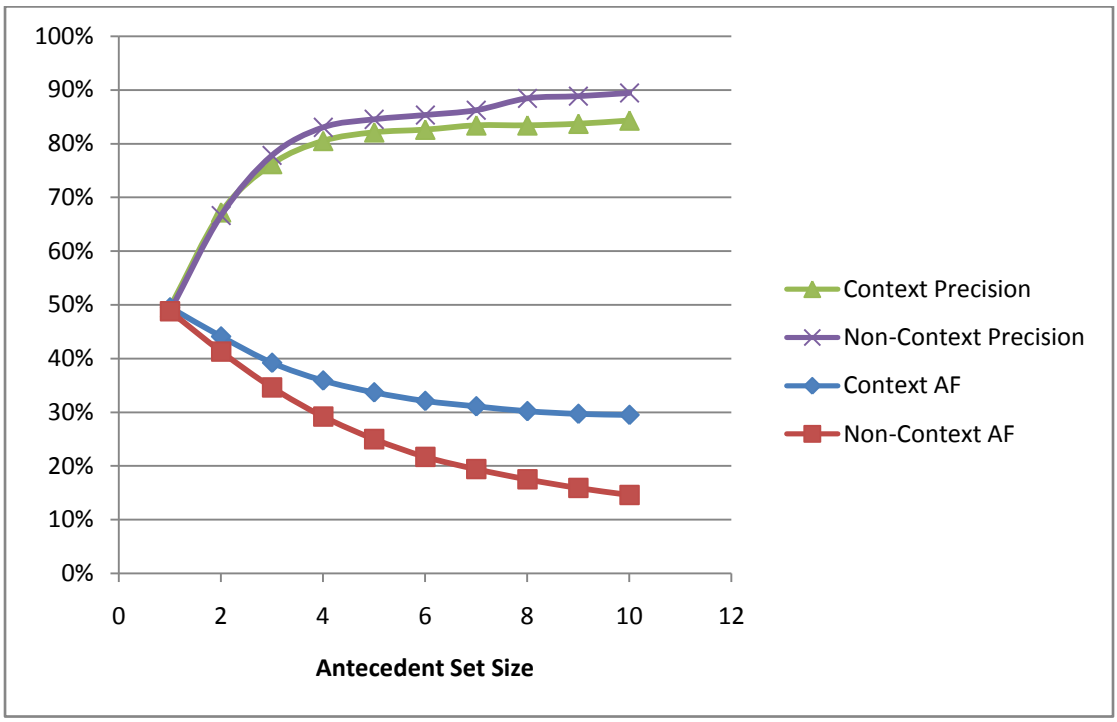


Figure 5.9.2– Contextual vs. Non-contextual resolution for various set sizes on the Wolverhampton corpus

The graph in Figure 5.9.2 shows the *AF* of the reference algorithm as it is allowed to include more antecedents in its set of candidates. The bars show the precision, which

measures the number of sets where the true antecedent appears (as described earlier). As the set size increases, the likelihood of the true antecedent appearing increases, improving the precision. However, this comes as a trade-off to *AF*, since the larger the set size, the lower the *AF*.

The results show that contextual segmentation keeps the *AF* from degrading beyond a certain point, whereas the non-contextual algorithm continues to be diluted by further antecedents. Segmentation limits the set size by the contextual boundaries and as the set size increases more anaphora will hit this boundary, which determines the asymptotic *AF* curve on the graph. In the non-contextual case, there is no limit so the *AF* appears more linear.

The effects of contextual segmentation are more pronounced in anaphora that appear in short contexts, or earlier in the text of longer contexts. These cases tend to comprise a smaller percentage of a document, and for this reason the effects are mitigated to some degree over the entire corpus. The 20% range for context *AF* shown on the graph quantifies this mitigation. A set size ranging from 1 to 5 antecedents shows the steepest decline in *AF*, which means that fewer of the cases benefited from segmentation. Candidate antecedent sets larger than 5 begin to be affected by the contextual segmentation, showing that the average distance to a contextual boundary is greater than 5 antecedents.

The difference in the precisions scores for the two cases shown on the graph approximates the number of cases where the antecedent crosses a contextual segment boundary. Using contexts the antecedents are limited to just those within the context. The non-context version is limited only by the set size proscribed in the test. Although the difference is small between the precision scores there are several factors which may contribute to a higher score for the non-contextual algorithm.

First, the algorithm does not account for cataphors, so cataphors, although appearing within a context, will not be found by the contextual algorithm. The same is true for the non-contextual algorithm, however it may eventually find a related earlier antecedent if the set size reaches far enough back.

Next, even without considering cataphora, if the base algorithm misses an antecedent due to an algorithmic or other implementation error, both precision scores will be penalized. However, the non-contextual version may, again, stumble upon an earlier related antecedent if the set size becomes large enough. Because the scoring method uses equivalence classes, these cases are not detected and instead counted as correct for the purpose of calculating precision.

Finally, there is the case where the antecedent truly lies outside the contextual segment. An analysis of the corpus reveals that these cases are actually quite rare and should not significantly affect the precision score. Out of a total of 1376 instances of pronominal anaphora in the corpus, just 13 cases (~1%) were observed to refer to antecedents outside the pronoun's contextual segment.

5.10 Summary and Conclusions

In this chapter I have examined the contextual information which can be derived from document structure and syntactic parsing. I have proposed a general representation which applies recursively to both and demonstrated how this can be integrated in the CAMEO text representation.

The chapter focused on one application of contexts in text processing, namely reference resolution. I discussed several issues that arise with contextual reference resolution, and I presented an analysis of first and second person pronouns in relation to embedded contexts, specifically dialog. I showed how processing and representing contextual information can produce a correct analysis of these constructions.

The more difficult problem of third person anaphora requires a deeper analysis, including a consideration of lexical and compositional semantics, which is outside the scope of the current research. Instead, I focused on the application of contexts using shallow syntactic methods. I argued that a purely syntactic approach to reference resolution will always be limited by semantic ambiguity, and should therefore be viewed as a pre-processing stage for filtering likely antecedents. With this view, the notion of *AF*, which incorporates both the antecedent set size and the true antecedent rank, becomes relevant.

For certain genres and styles of text, contextual processing can show modest gains on third person anaphora. (Larger gains are achieved for first person anaphora, but the application is limited to an even smaller percentage of documents). Using a document comprised of short stories and literary excerpts containing dialog, I showed that using contexts improves the *F-score* and *AF* of a baseline reference resolution algorithm for all antecedent set sizes. The gain would likely become more pronounced if the contextual structure was considered by a following stage of semantic constraint processing.

Finally, I demonstrated a further application of contexts to reference resolution using contextual segment boundaries to limit the number of antecedents suggested by the reference resolution algorithm. Comparing this to an arbitrary size showed that *AF* can be kept asymptotically within a higher range. Because only a small percentage of the text encounters the segment boundaries at smaller set sizes, the advantage increases with the size of the set.

A higher set size was shown to increase precision but decrease *AF*. A lower *AF* translates into higher ambiguity and complexity for the antecedent set, which would lower the performance of a constraint processing module. Using segment boundaries helps keep the *AF* at a higher level without significantly impacting precision.

6

Symbolic and Distributional Methods

In this chapter, I will briefly discuss strategies for augmenting natural language processing with both symbolic and distributional information. I will show how these two different approaches can be combined and how the CAMEO representation facilitates both methods. Combining distributional information with symbolic processing techniques is a relatively novel approach in computational linguistics. The two methods are somewhat orthogonal and in some sense address separate problems, or at least separate models. Because of these differences an augmented approach is warranted, where distributional information is used to enhance symbolic techniques.

In the introduction I explained how research into distributional methods has increasingly made use of symbolic information. Incorporating symbolic information into distributional studies is nothing new. Shallow syntactic information was incorporated into early distributional experiments, as for example Hindle (1990), who used syntactic verb-object dependencies to determine word similarities. However, research in the 1990's placed an emphasis on statistical similarity measures based solely on collocative information (e.g. Periera et al., 1993; Waterman, 1996; Schütze, 1998; Li and Abe, 1995). At the same time, distributional methods of semantic extraction were being developed as an alternative to manual methods (e.g. Hearst, 1992; Light, 1996). Recently, efforts to combine these two approaches have emerged. Statistics about the lexico-syntactic patterns and relationships a lexeme participates in have been used in conjunction with raw collocational statistics to refine traditional statistical methods.

Lin, et al. (2003) acknowledge the limitations in deciding synonymy using strictly collocational evidence. They point out that lists of distributionally similar words often

contain antonyms or members of a multi-valued semantic category (e.g. color adjectives), rather than strictly synonymous words. The insight here is that semantic similarity is not the only property which causes distributional similarity. Other semantic properties such as antonymy may be at work, diluting statistically derived word clusters that rely solely on collocational information. (Chapter 6 gives a treatment of antonymy in distributional experiments).

The solution proposed by Lin et al. is to refine a word cluster using heuristic lexical patterns that have been determined to correlate with a semantic property. The examples used by Lin et al. to filter antonyms are the patterns [**from** *X* **to** *Y*] and [**either** *X* **or** *Y*]. These patterns are conjectured to mark the lexemes *X* and *Y* as semantically incompatible. Lin et al. report a high precision and recall when deciding synonyms vs. antonyms using these patterns on distributionally similar words.

Going beyond lexical patterns, efforts to utilize some syntactic information have also been applied to distributional semantic classification. Hindle (1990) and Lee (1999) are examples of distributional approaches using a specific syntactic relation (verb / object pairs). Grefenstette (1992) and Lin (1998) incorporate more general dependency relations into a word's context vector, but only as a means of refining the collocational information associated with a word.

Padó and Lapata (2003) go one step further and attempt to define a general distributional approach that considers both lexical and syntactic information in a parametric framework. They argue that the semantic space described by vectors of strictly lexical collocations conflates the important contextual information of word events. They suggest that linguistic information should be included in the vector space model, and formalize this idea by generalizing over dependency relations. Rather than creating unordered vectors of collocations (i.e. bags), they propose a weighted vector of dependency relations. In this scheme, the encoding of dependencies is done in such a way that dependencies beyond simple head-modifier relationship can be included in the distributional information. In addition, the inclusion of parametric weights allows linguistic information to be leveraged, since certain dependencies can be made to have a higher significance than others.

The results reported by Padó and Lapata and others confirm that dependency information can be a useful feature for distributional similarity measures. One of the challenges to this approach, which has made it difficult in the past, is obtaining the dependency information. It has only been relatively recently that large multi-million word corpora with syntactic dependency information have become feasible, and these resources can still be expensive to produce. Furthermore, extracting the dependency information from these types of resources for use in distributional experiments also involves some effort.

My goal in this chapter is simply to point out the intrinsic support in CAMEO for these methods, and to provide some context for the following chapters which focus in depth on distributional processing. I will begin by looking at the kinds of distributional information that can be derived from the CAMEO representational forms. Next, I will explain the issues involved with integrating distributional information that has been derived externally. Finally, I will propose one application of distributional techniques for symbolic processing which forms the basis of the experiments in the final chapter.

6.1 Distributional Information in the CAMEO Representation

One of the stated goals of the CAMEO representation was to facilitate the collection of distributional information using the intrinsic representation. Several properties of the representation aid in this respect. The use of globally unique identifiers on classes and lexemes simplifies distributional queries. In Chapter 3, I briefly noted that class information is accumulated during processing. Every common noun encountered during text processing is represented as a class and given a unique identifier, which is referenced in the representation language. This allows a simple means for distributional analysis to be performed by processing instances where the class id appears.

Although only common nouns are given explicit class identifiers, any lexeme processed in the system can be used to collect distributional information. All word tokens from open-class categories are indexed by lexeme, and closed-class words (determiners, verbal auxiliaries, and certain quantifiers) can be queried directly as

attribute values. For example, to find all co-occurrences with the determiner *the*, a query could be run for all elements having an attribute [DET = *the*].

A further feature of the representation that aids distributional processing is the explicit representation of objects. Just as distributional information can be collected for a class lexeme, an object identifier can be used in distributional queries to find collocational and dependency information. This has advantages over other representations, such as PC, where the notion of an entity is less explicit and would require further processing beyond a simple query to recover distributional information. For example, in CAMEO the identifier of an object which has been resolved to an equivalency class can be used in a simple global query to collect distributional information for the entire equivalency class. Figure 6.1 gives pseudocode and an actual XPath expression which would filter all objects that are members of an equivalence class containing **id**(*x*).

Pseudocode:

```
for all obj remember id(this) then do
      if there exists eq containing both id(this) and id(x) do …
```

XPath:

```
<xsl:template match="//obj[//eq[ obj[ @IDREF='x' ]/obj/@IDREF=@ID ]"/>
```

Figure 6.1

Example distributional query for all objects related to **id**(*x*) **t**hrough some equivalence class **eq**

Note that distributional information which was collected before the equivalence class was created is logically aggregated by a coreference resolution operation. Each new referent which becomes a member of the equivalence class contributes any distributional information it might have, and at the same time inherits the distributional information of the group. Enforcing coherence of the distributional information could possibly provide another dimension of constraint processing for the coreference task.

The encoding of syntactic structure in the representation means distributional information beyond simple collocations can be recovered directly. For example, it is possible to determine the number of times the noun *gun* appears as the object of a preposition by forming a query on all **rel** elements where the OBJ attribute has the class identifier *gun*. This same technique can also be used to find other direct distributional information, such as modifiers, verb forms, etc. Information collected in this fashion can be made available to other modules by annotating the class element of the given noun.

As more syntactic and symbolic information is incorporated into distributional processing, more sophisticated search patterns must be supported by the representation. For example, Levy and Andrew (2006) report on a query language created for syntactic tree structures which allows detailed co-occurrence patterns to be applied to a corpus. In CAMEO, these types of patterns, and more indirect and variable distributional patterns, can be applied through the flexible query language (XSLT) which operates on the representation.

For example, it may be advantageous to collect distributional information for a compositional phrase, such as [***x*** *be* ***y*** *fault*], where ***x*** and ***y*** represent any type of syntactic construction, from simple clauses [*The failure was the engineer's fault*], to more complex phrases [*The judge did not believe the crime to be the fault of the victim*]. To allow complex distributional queries such as this a representation needs to allow a simple means to encode the core information. In CAMEO the representation would appear as:

ctx[
  obj[ID=*n* **class**[*fault*]]
  evt[ACTION=*be* O=*n*]]

The **ctx** element serves to denote the group of elements included in the phrase under study and as a suitable container for annotation. Each element in the distributional query contains only the minimum information required to describe the construction being investigated. This concise representation of the relevant constraints will match a wide range of constructions containing the desired phrase. A distributional query built from these constraints and applied to the representation will be able to return more than

just direct instances of the query. Many different syntactic variations can indirectly match the core constraints and satisfy the distributional query. Additonally, a dependency query (like that illustrated here) can also serve as a dependency rule when it is cast to designate some property to matching contexts. Chapter 9 uses this type of dependency query in rules that designate the semantic classes of nouns.

Although distributional processing is certainly possible with any representation (whether through ad-hoc approaches or transformations into database formats), the global indexing of lexemes and the explicit representation of objects in CAMEO, together with the flexible query language provided by the implementation, allows the integration of distributional methods directly. This includes the possibility for including deep syntactic features in conjunction with more shallow collocational information. Additionally, because the representation is implemented using XML, many existing XML processing tools and utilities can be applied directly to the representation (e.g. Apache Xindice, Berkely DB XML, IB Search Engine, eXist-db, Tamino XML Server, etc.).

6.2 External Resources

The processing model of CAMEO, which adopts the XML DOM, allows a general means for adding attributes or other elements to an entity in the representation. Information appended in this manner is adjunct to the representation and becomes accessible to all modules operating in the system, providing a form of annotation that extends the representation to accommodate augmentative symbolic and distributional methods. Although representing external information this way is not strictly necessary (since a processing module may have its own interface to the external source), it serves to merge the two sets of information conveniently. By incorporating the external information directly into the representation, processing can be more efficient. Attributes and information which are expensive to extract using external resources can be processed once and reused by subsequent modules.

This scheme of annotating the representation is flexible and accommodates both symbolic and distributional information because the form is not constrained. Some

examples of external symbolic resources that might be used to augment linguistic processing include lexical and semantic resources such as Machine Readable Dictionaries (MRDs), Lexical Knowledge Bases (LKBs), or general world knowledge. For example, the WordNet semantic taxonomy (Fellbaum, 1998) could be used to annotate the representation with semantic attributes or sense information (as described in the next section). By transferring this information into the representation, linguistic tasks that run in the system can access this information directly from an element in the representation, rather than having to extract it for each element themselves.

Like external symbolic information, representation of external distributional information is flexible and can take several forms. In each case the representation provides a framework for the external information, leaving the interpretation for independent processing modules. Each element in the representation can be expanded to take distributional information in the form of word vectors or discrete statistical information. The lexical entries of the lexis context can be expanded to take lexeme-based distributional information, and entries in the **classes** context can be annotated with distributional information for differentiated lexical senses.

For example, one approach to adding distributional information to a class entry is

```
classes [
        classdef[  ID=c1 LEX=123
              collocate[  LEX=1403 COUNT=812  ]
              collocate[  LEX=1507 COUNT=12456  ]
              ... ] ]
```

where the collocation information is comprised of simple context pairs from a specific dependency relation (e.g. adjective / noun), and may have been culled from an external corpus implemented in a different representation.

Distributional information which includes dependency relations can be represented using the standard CAMEO elements as child elements to the distributional anchor. For example, dependency information from an external corpus implemented in a GR representation can be added to the **classes** context by translating the GR dependencies to CAMEO elements and adding them as child elements of the **classdef** element used to

anchor the distributional query. Assuming the distributional processing included a means of aggregating the information, this would produce a representation similar to

classes [
classdef [ID=*c2* LEX=*149*
rel[PREP=*137* COUNT=*53* OBJ **]**
evt[ACTION=*184* COUNT=*39* S **]**
evt[ACTION=*122* COUNT=*53* O **]**
mod[LEX=*188* COUNT=*312* **]**
…]]

where attributes in the distributional elements that are unspecified are unconstrained. The role of the anchor in each distributional element is indicated using a defined but empty attribute. In this example, the class is annotated with several distributional relations. The first **rel** element represents 53 instances where the class appears as the object of the preposition **id**(*137*). The next **evt** element represents 39 instances where the class **id**(*c2*) appears in the subject position of a verb phrase incorporating **id**(*184*) as the head verb. Note that for this **evt** entry the form of the verb phrase is unconstrained in the representation, and thus aggregates passive, modal, and tense variations in the distributional data.

External distributional information can be leveraged at various stages of processing for a range of applications. For example, during QA processing the framework might return multiple answer candidates for a given query. Distributional information could be used to score semantic similarity for words in the answer candidates against words in the question, providing a means of ranking the answer candidates. Alternatively, if a query results in no answer candidates, the distributional information could be used in expanding the query.

For text generation, distributional information might be used in discourse planning. Statistical information about the frequency of words or co-occurrence would serve to help select surface syntactic forms that conform better to natural speech patterns. Distributional information could also be integrated with the surface text generator itself,

by boosting or penalizing certain syntactic forms based on statistical evidence (e.g. fronted direct objects).

As a final example, consider the coreference resolution task. Here the distributional information can be used to augment the measure of a coreferent's compatibility. This could take the form of disambiguating word senses, boosting based on high statistical collocative evidence, or measuring semantic similarity, as the following section will show.

6.3 Distributionally Derived Attributes

To give an example of how symbolic and distributional techniques can be combined to augment a language processing application, in this section I will propose a technique designed to enhance the coreference resolution task presented in Chapter 5. Recall the coreference task from Chapter 5 consisted of determining the set of antecedents for use in a ranking algorithm to determine coreference. The experiments demonstrated an implementation of a basic reference resolution algorithm, based on Hobbs (1978). Like the Hobbs algorithm, most coreference resolution algorithms include as a basic step the elimination of incompatible antecedents, where incompatibility is usually determined from a simple match over a small number of attributes.

For example, in section 2.2.1.2 I listed several attributes which can be attached to objects: *gender, animacy,* and *plurality*. When these attributes appear with an object instance, the coreference resolution algorithm is better able to filter incompatible candidates for referring expressions, improving precision and (indirectly) recall.

A good source of attribute information of this kind is a lexical resource such as WordNet. The WordNet semantic hierarchy of nouns allows coarse grained attributes (like those listed above) to be determined for most common nouns. One method of determining this would be to trace the hypernymy relation of a noun back to an ancestor node which is considered the source of a particular attribute. For example, the noun *thought*, can be traced back to the *cognition* root node, and therefore deemed to have the *inanimate* attribute.

As with most lexical techniques, using WordNet to determine nominal attributes suffers from polysemy. A noun appearing in the WordNet database will very often contain multiple senses, each with a potentially different semantic ancestor. Using the wrong sense can potentially lead to assigning an incorrect attribute to a noun, degrading the accuracy of the coreference resolution algorithm. However, this effect is mitigated slightly by two factors. First, the coarseness of the attributes in question results in many senses mapping to the same attribute. For example, the noun *plane* has five senses in WordNet, but each maps to the *inanimate* attribute. Second, the senses in WordNet are generally listed in order of frequency, such that the first sense is the most common. Using the first listed sense of a word will result in the correct attribute on average, although this is highly dependent on the corpus being analyzed.

One of the limitations of this type of symbolic technique is lack of complete coverage. Nouns that are not listed in the lexical resource cannot be processed in the same manner. For instance, the noun *F-14*, which refers to a type of military aircraft, is not listed in the WordNet database, although it occurs many times in the MUC-7 corpora. In order to assign attributes to unknown nouns, a method for determining their semantic properties is needed, i.e. lexical acquisition.

In Chapters 7 and 8, I will investigate statistical approaches to semantic similarity using large-scale distributional information. These techniques are well-suited to lexical acquisition in large corpora containing multiple documents, where a high number of instances (i.e. many thousands) can be observed to smooth statistical aberrations. In smaller corpora such as single documents, these techniques must be adapted for the much smaller number of occurrences (i.e. on the order of tens or less). As I discussed earlier, integrating more symbolic features into a distributional approach can be used to adapt these techniques to a smaller corpus. (Chapter 9 explores this idea.)

For example, a common approach to collecting large-scale distributional information is to use an *n*-word window of collocations centered on the word under study. In a small corpus, where the word under study might appear only a few times, this can produce a very small feature vector with mostly unique tokens that is ill-suited for statistical manipulation. One alternative is to use more symbolic information for distributional features, which can be more reliably aggregated, such as an object's role in a verb or prepositional phrase. This type of information is salient in a topical discourse because

unknown nouns are often anaphoric with known nouns and exhibit parallel syntactic construction. Returning to the example unknown noun *F-14*, the second MUC-7 document contains two parallel syntactic constructions with one of its coreferents (*fighter*): both appear as the object of the verb *crash*, and both appear as objects of the preposition *for*. Using these data for semantic similarity in a large-scale corpus would most likely be specious, yet for a topical small-scale corpus this type of information can prove significant. A simple similarity measure based on the distribution of common symbolic events can then be used to associate an unknown noun with a known noun, and the attributes of the known noun can be adopted for the unknown noun.

6.4 Discussion

The goal of this chapter has been to suggest how symbolic and distributional information can be integrated to enhance traditional approaches to language processing. I have tried to show how the CAMEO representation supports the collection and annotation of distributional information intrinsically, as well as simple methods for annotating the representation with externally derived distributional information. The noun class context is a repository for this information and supports external task-specific information, as well as data collected and processed from the representation itself. In addition, distributional information can be attached to discrete entities (i.e. **obj** elements) using attributes. Using the internal properties of the representation simplifies the implementation of both symbolic and distributional processing, operating in a complementary arrangement, to augment tasks such as coreference resolution. As new methods of applying distributional processing to complement symbolic tasks are developed, this will become an increasingly critical property of a general framework.

As I discussed in the introduction, distributional processing is most often employed in resolving ambiguity using probabilities computed from statistical frequencies derived from large-scale corpora. This approach has been successfully applied to syntactic tasks like tagging and parsing, and to a lesser extent to lexical tasks such as word sense disambiguation. Where there exists distinctly ambiguous choices that can be labelled and counted in some way, these techniques can model probabilities adequately. For example, the frequency information which is used to order the word senses in WordNet approximates the probabilities of the individual senses. On the other hand, applying this

approach directly to problems not having clearly ambiguous choices is less intuitive. For instance, coreference resolution is a type of disambiguation but applying distributional approaches directly to the problem is not necessarily helpful. However, there are many possibilities for integrating distributional information with symbolic processing which may be applied indirectly to a task.

In this chapter I have suggested one such possibility using a distributional measure of lexical similarity to derive semantic attributes for unknown class nouns, which are then used indirectly by the coreference algorithm to qualify candidate referents. Other indirect applications of distributional processing to these kinds of tasks may address different aspects of lexical ambiguity.

In the next two chapters I will address the general properties of statistically based similarity measures employed in distributional processing, and develop an adaptation that can improve their applicability as adjuncts to language tasks in smaller corpora. I will refine these techniques in Chapter 9 to extend the ideas proposed in this chapter, by deriving symbolic rules for determining semantic attributes. I will also address several of the limitations with the approaches presented in this discussion.

7

Statistical Similarity Measures in Lexical Acquisition

In Chapter 6 I showed how distributional methods which use internally derived distributional information can be implemented using the CAMEO representation. I also explained how distributional information from external large-corpus processing could be used to annotate the representation for use in conjunction with other language processing tasks. As I pointed out earlier, augmenting the representation with this statistical information facilitates probabilistic methods that complement the symbolic processing in the representation, e.g. disambiguation strategies. In this chapter I will look closely at the kinds of external distributional methods that can produce this information, by evaluating several typical approaches. (I refer to these methods as "external" with respect to a language processing application because they use a separate (and typically much larger) corpus to obtain their results.) As mentioned earlier, large-corpus distributional processing is best-suited for lexical tasks and this discussion will only address lexically scoped processing, in particular lexical semantic acquisition which is a useful characterization of the application of distributional processing.

Because this chapter is mainly concerned with evaluating several existing well-known distributional techniques which are based on collocative features, I will not consider the relatively recent application of syntactic dependency features or other symbolic adjuncts, as discussed in Chapter 6. In addition, I will gloss the details of the experimental framework, since it is not relevant to the discussion. The focus here is not the representation and implementation of these methods, but rather the properties of the results and the qualities of the algorithms used to achieve them.

I will begin by introducing lexical semantic acquisition and statistical similarity measures, which can be used for lexical semantic acquisition. Next, I will review the obstacles faced when implementing large-corpus distributional algorithms. I will then discuss the difficulties that arise when attempting to objectively evaluate statistical similarity measures (namely, the lack of a gold standard). To address this problem, I will look at some lexical properties of adjectives which make adjective antonyms uniquely suited to be used in evaluating the performance of statistical similarity measures. Finally, I will present the results of experiments using three distributional approaches to semantic acquisition of adjectives, using the proposed antonyms for evaluation.

7.1 Lexical Semantic Acquisition

Semantic acquisition refers to automated methods that can discover useful semantic features of linguistic objects, versus manual methods that require human intervention. Manual methods are expensive to implement and difficult to validate and are thus less desirable, although it is sometimes necessary to use them. For instance, symbolic methods that deeply analyze linguistic data have proven to be difficult to automate, requiring large-scale symbolic databases to be built using hand-coded methods instead (e.g. Cyc (Lenat, 1995), WordNet (Fellbaum, 1998)).

Some effort has been made at augmenting these manual methods with semi-automated techniques. Hearst (1992) describes a method for automatically discovering hyponym relations using surface cues in unrestricted text. Using high-confidence lexico-syntactic constructions, Hearst demonstrated how hyponyms can be mined from large corpora. Although the hyponyms are extracted automatically, this is considered a semi-automated method because the lexico-syntactic patterns must be determined manually.

The method employed by Hearst is related to the work of extracting lexical relations automatically from MRDs by searching for specific syntactic surface patterns and cues in word definitions. This has been actively studied in the literature (e.g. Richardson et al., 1998) and generally produces fine-grained results, but is sensitive to the limitations of the word definitions (e.g. omissions, polysemy, circularity, etc.).

Light (1996) applied a similar approach to morphology. Using hand-coded morphological rules he was able to acquire fine-grained semantic features of various parts of speech. He argues that surface cues such as morphology are generally accurate, abundant and reliable. However, these types of methods, in addition to requiring manual coding and analysis of the rules and surface patterns, only provide a partial solution to lexical acquisition.

In contrast to these pattern-matching, semi-automatic approaches to acquisition, distributional methods can be implemented using fully automated statistical techniques, processing large amounts of data to uncover statistical features and probabilities. For instance, the probability that a verb selects for a certain noun might be estimated by counting the number of co-occurrences of the noun and verb in proportion to all occurrences of each.

The hypothesis underlying these statistical approaches is that words that have similar semantics (or syntax or properties), will have similar distributions. By measuring the similarity of the distributions, the similarity of the words can be induced, and semantic properties can then be automatically acquired.

One semantic property commonly derived from distributional approaches is synonmy, or class membership. Knowing the degree of similarity among groups of words and/or lexical classes enables the classification of unknown words (i.e. lexical acquisition), or construction of new lexical classes. A word's distributional profile then becomes a measure for association with groups of words forming a semantic class. Thus, comparing the distributions of words becomes a way of measuring semantic similarity.

7.2 Distributional Approaches to Semantic Similarity

There are two major aspects to consider when implementing distributional approaches to semantic similarity: 1) the types of distributional features that are to be used (e.g. collocations, syntactic relations, etc.), and 2) the algorithm for computing the similarity metric. The success of an application will largely depend on these two design decisions.

7.2.1 Features of Events

An *event* in a distributional experiment is an observed occurrence of a certain lexical pattern. The features of an event are other bits of information about the occurrence. There are many different kinds of features that can be extracted from an event including syntactic, contextual, and morphological forms. Features ideally are symptoms of linguistic principles, but many turn out to be spurious.

Syntactic Features

Syntactic features are taken from the syntactic construction of the text. These normally require some sort of parsing to recover and include dependencies such as head-complement relations. Other examples of syntactic features include information on constituency (the syntactic constituent a given word participates in), whether the event occurs in a clausal component, or whether the event is part of a conjunction.

Contextual Features

Contextual features are what is usually thought of when designing distributional experiments. Specifically, word *n*-grams record the distributional context of a word, without regard to more complex processing such as parsing. But contextual features do not need to be limited to the immediate context of a word. Features can sometimes include sentence level, and even document level contextual information. For instance, the most frequent noun or verb for a given document might be recorded along with each event.

Semantic Features

Semantic features are the most difficult to obtain and utilize, as they require symbolic processing and existing lexical resources. A typical use of semantic features is hierarchical class smoothing, where individual words are smoothed into larger classes to alleviate data sparseness. Other possibilities include semantic properties such as the level of polysemy, or the existence of antonymy. Note an existing semantic resource would be necessary to implement any of these examples.

Surface Features

Surface features are any other bits of information about the event that might be gleaned. Some examples are morphology, punctuation, and alternations. In some cases these turn out to carry significant information. For instance, a preceding comma may turn out to be a good predictor of the sense of a polysemous word.

7.2.2 Similarity Measures

Similarity measures are a core component of unsupervised statistical approaches to NLP. For example, clustering techniques use similarity measures by calculating the "distance" between objects (or classes). Clustering has been applied to NLP for tasks such as word sense disambiguation (Brown et al., 1991; Chen and Chang, 1998; Dagan et al., 1995; Dolan, 1994; Pederson and Bruce, 1997; Schütze, 1998), inducing semantic classes (Lapata and Brew, 2004; Hindle, 1990; Hatzivassiloglou and McKeown, 1993; Merlo and Stevenson, 2001; Periera et al., 1993; Waterman, 1996; Li and Abe, 1995), and learning syntactic properties (Brill et al., 1990; Finch, 1993; Schütze, 1995).

Similarity measures rely heavily on information theory and other well developed techniques from machine learning. In this section I will examine three similarity measures that have been proposed in the literature. For convenience, I will label these as: Minimum Mutual Information (MMI) (Hindle, 1990), Tau Coefficient (TAU) (Kendall, 1938; Hatzivassiloglou and McKeown, 1993) and Distributional Clustering (DC) (Periera et al., 1993). Each of these measures uses a very different approach to determining the similarity of distributional data. I will first present each similarity measure in some detail, and then discuss evaluation strategies.

The original experiments presented in the literature to demonstrate these similarity measures used various forms of lexical relationships for the distributional data (e.g. *noun-verb, adjective-noun,* etc.). In order to compare the similarity measures, it is helpful to generalize the form of the distributional data. For a given word *w* we define a vector v_w of *events* representing selected collocations of *w* observed in the corpus. An event describes a word token and the corresponding count of observed collocations (i.e.

frequency) with *w*. The elements of the word vector v_w comprise the list of all events involving the word *w* observed in the corpus. The following discussion uses this terminology to explain the mechanisms of the three similarity measures under study.

7.2.2.1 Minimum Mutual Information

Minimum Mutual Information (MMI) was proposed by Hindle (1990) and is based on a variation of mutual information from information theory. Mutual Information (MI) provides a measure of the information of a joint event, using the joint and independent probabilities of those events.

In brief, Hindle (1990) defines the similarity of two nouns by comparing the (estimated) MI of verb events that appear in common. Because MI is calculated using a logarithm, very small ratios (representing less information) will be negative, and larger ratios (representing more information) will be positive. When the sign of the MI for a verb event agrees between two nouns, the two nouns are hypothesized to have a similar semantic relationship with the verb. In these cases, Hindle selects the minimum absolute value. The sum over all such cases is taken as the measure of similarity between the two nouns.

The algorithm, in more detail, begins by calculating an estimate of MI for each element in a vector's distribution using frequency information. Hindle calls this a co-occurrence score and it is given by

$$C(v,e) = \log_2 \frac{\frac{f(v,e)}{N}}{\frac{f(e)}{N}\frac{f(v)}{N}}$$

where *v* is a vector containing element *e*, *f*(*v*, *e*) is the frequency of event *e* in vector *v*, and *N* is the total number of events in all vectors.

For two vectors being compared, if a given element has a concurring sign in both vectors, the *minimum* magnitude of the two values is added to the similarity score. The co-occurrence score *C* is produced by summing in this manner over all elements.

Formally, we define the similarity score of two vectors v_1 and v_2 as the Minimum Mutual Information (MMI) shared between the two vectors, given as

$$MMI(v_1, v_2) = \sum_{i=1}^{N} C_{\min}(e_i, v_1, v_2)$$

where e_i is a shared element of vectors v_1 and v_2 and $C_{\min}$ gives the minimum co-occurrence score of the element as

$$C_{\min}(e_i, v_1, v_2) = \begin{cases} \min(|C(v_1, e_i)|, |C(v_2, e_i)|), C(v_1, e_i) \times C(v_2, e_i) > 0 \\ 0, \quad otherwise \end{cases}$$

From this we see that two word vectors that have no events in common will have a MMI similarity score of 0. Also two word vectors whose co-occurrence scores always differ in sign will score 0.

The intuition behind this approach is that two word vectors that are semantically related will have significant co-occurrence scores on the same elements, since they should produce similar distributions. If there is no semantic correlation, the co-occurrence scores will instead be mismatched. The correlations reinforce the similarity score, and the mismatches are ignored.

7.2.2.2 Tau Coefficient

The Tau Coefficient (TAU) was proposed by Kendall (1938) and employed by Hatzivassiloglou and McKeown (1993) in their work on automatically identifying adjective scales. The Tau coefficient uses the *differentials of events* as a means of comparison. It measures the similarity between two vectors by counting the number of event differentials whose sign concurs across the two vectors (concordances), subtracting the number that do not (discordances). For differentials that are equal there is no effect.

To calculate the Tau coefficient, the elements of a vector are exhaustively enumerated as unique pairs. The differential of each pair is calculated by subtracting the element frequencies, and the sign is noted. The results are used to compare with another vector. The differential sign of every corresponding pair in the two vectors is compared.

If the signs agree, the pairs are said to be concordant. Signs that differ are said to be discordant.

The Tau Coefficient is defined as

$$\tau = p_c - p_d$$

where p_c and p_d are the probabilities of a concordance and a discordance, respectively. Tau can be estimated using

$$T = \frac{C - Q}{\binom{n}{2}}$$

where n is the number of elements in the vector, and C and Q are the numbers of observed concordances and discordances, respectively. From this we see the range of the Tau coefficient to be -1 to +1, where +1 indicates strong similarity, -1 indicates strong dissimilarity, and 0 indicates no correlation.

The Tau coefficient is intended to capture the proportional "shape" of the distribution and disregard the absolute quantities. This is important for corpus based work because the frequency information is only approximate. Only the relative likelihood of an event compared to another event is significant in this approach. If two word vectors have the same event more likely in relation to another event, it is possible that this is due to the same semantic property. The Tau coefficient attempts to capture this in its similarity score.

It should be noted that because this metric must calculate the differential for every pair of elements in every vector, it is too computationally expensive for large vectors. For class based probabilities with modest numbers of elements, such as those used in these experiments, it becomes feasible.

7.2.2.3 Distributional Clustering

MMI and TAU are both direct similarity measures between two discrete vectors. In contrast, Distributional Clustering (DC) is a soft-clustering technique that measures similarity between *clusters* of vectors. DC, as presented by Pereira et al. (1993),

incorporates the Kullback-Leibler distance between clusters in a divisive clustering scheme derived from simulated annealing techniques. The Kullback-Leibler distance between two distributions (i.e. vectors) is given as

$$D(p\|q) = \sum_x p(x)\log\frac{p(x)}{q(x)}$$

Pereira et al. used the Kullback-Leibler distance in conjunction with a divisive clustering procedure that essentially creates a set of semantic sense classes (represented by clusters). The centroid of a cluster is a derived vector that gives a hypothetical "prototypical" distribution for the sense over all events. A word vector (which may conflate multiple semantic senses) is interpreted as a probabilistic distribution over these senses (i.e. clusters)

In the DC algorithm, the Kullback-Leibler distance is used in the re-estimation of the cluster centroids to find the distance between an actual word vector (observed in the corpus) and the estimate given by a cluster centroid. A cluster centroid is calculated as the average of all word vectors, weighted by the simulated annealing "temperature". (Since the cluster centroids are derived from an average of the actual word vectors, there is no issue with elements having a value of zero in the KL denominator, a problem often encountered when applying this distance function.) A high temperature gives more weight to local word vectors, producing a more localized centroid.

Once the centroid has been determined, the *distortion* of a cluster can be measured by calculating the (KL) distance from the centroid to each word vector (subject to the temperature weighting). The distortion gives a measure of the semantic focus of the cluster. When the distortion is low, the word vectors belonging to the cluster are relatively close (where "belonging" means having the most weight).

Since neither the word vectors nor the centroids can actually "move" (the distributional statistics are static), the centroids are adjusted by selecting which word vectors are associated with the cluster (using the temperature). By changing the member vectors, the average of the vectors will change and this determines the location of the centroid. The goal is to find centroids which truly represent semantic senses, and this amounts to selecting the best groupings of the observed word vectors.The re-estimation of the cluster centroids is achieved by minimizing the individual cluster distortions

(measured against the observed word vectors in the corpus), while simultaneously maximizing the overall entropy of the system. This is accomplished by minimizing a "free energy" function, which incorporates both objectives. The free energy includes a parameter analogous to temperature in deterministic annealing. Increasing this temperature parameter gives more influence to vectors close to a cluster's centroid. A high temperature in the limit would produce a cluster for every vector, with the centroid equal to the vector.

The algorithm begins with a single cluster and very low temperature, which gives all vectors equal representation and produces one centroid equal to the average of all vectors. The algorithm then iteratively splits each cluster centroid in two, using small random perturbations, and increases the temperature until the re-estimation function causes the two centroids to diverge. The algorithm can be stopped when the desired number of clusters is found.

7.3 Obstacles

There are several obstacles to overcome when designing distributional methods of acquisition. One difficulty is data sparseness. Although data sparseness affects many methods of language processing, it is particularly acute for distributional techniques which depend on reliable frequency and probability information. Another obstacle is polysemy, which also afflicts many methods of language processing. Before moving on to evaluation strategies, I will briefly discuss these two issues and how I addressed them in my experiments.

7.3.1 Data Sparseness

Distributional similarity measures especially suffer from data sparseness issues. It is very probable that two word vectors, each comprised of hundreds of events, may have only a handful in common. This makes comparisons based on shared events less accurate and robust, since only a small percentage of events can be used for comparison. One solution to this problem is to use smoothing to collapse groups of events into classes based on some significant property. The most obvious property to use when reducing a word vector is the semantics of the events. If a set of words are

synonyms, near-synonyms, or semantically related in some way, they can be replaced by a single class which is more likely to be shared among other word vectors. In this manner a vector of hundreds of event elements with smaller frequency counts may be reduced in size, resulting in fewer event elements with larger aggregate frequency counts.

In the experiments that follow, I use this strategy with a pre-existing taxonomy of word classes to reduce the large distributional vectors to smaller class-based vectors. I utilized the WordNet 1.7 taxonomy to achieve this by tracing the hypernym relations for each word event (i.e. noun) in the vector back to its root ancestor, referred to in WordNet as a "unique beginner" (Miller, 1995). I also experimented with an alternate configuration using the second level of classes, i.e. tracing the hypernym relations for each event back to its penultimate semantic class. This is explained in more detail further on.

7.3.2 Polysemy

One of the more difficult problems when attempting to do automatic processing of natural language is polysemy. The distributional data available for these kinds of techniques do not usually include labelled sense distinctions for polysemous words. This affects not only the distributional profile for word vectors, but also other processing used to facilitate the operation, such as using the class information described above.

The disambiguation of word senses is itself an area of active research and a very difficult problem. Thus it is not feasible to disambiguate distributional information like that used in these experiments. The simplest and most common technique to work around this problem is to default to the most probable sense, if available. For the WordNet taxonomy used in these experiments, the first listed sense of a word is usually supposed to be the most common sense. This strategy was implemented in the first stage of the experiments and is detailed below.

Another approach proposed by Resnik (1993) is to characterize each event as an equal probability distribution over all senses of the word token. For instance, instead of a single class representing the primary sense of an event, an event would be represented

by a distribution over the classes related to all its senses. Each single word event is thus interpreted as a collection of equivalent fractional events over all senses.

Both of these approaches have been incorporated in the experiments that follow.

7.4 Evaluating Similarity

In order to compare the performance of similarity measures and judge their relative merits, an objective evaluation strategy is needed. However, finding such a strategy has proved difficult. In some experiments only a qualitative analysis of the end results is offered. Although a qualitative analysis is sometimes helpful to get an intuitive grasp of the characteristics of a similarity measure, it is difficult to make quantitative predictions about the performance based solely on this type of evaluation.

Intrinsic measures can sometimes be useful to evaluate the coherence of clusters, groups, or ranks built using similarity measures. These intrinsic metrics may also help in refining the parameters of an algorithm, but it is often difficult to draw conclusions about the extrinsic quality of the results. There is rarely an understood relationship between the types of intrinsic measures available and genuine linguistic properties. Thus, like qualitative analysis, intrinsic measures only provide a general intuition about the performance.

A more desirable approach is to devise a task-based evaluation, which allows a similarity measure to be evaluated indirectly through its effects on a real-world natural language application or task. This method requires much more effort but gives a better measure of the linguistic properties being affected by the algorithm. The difficulty lies in finding or creating an appropriate task. Even when a task is appropriate, it may not always be feasible to obtain labelled data, which is required for evaluation. Some researchers have turned to artificial tasks which use automatically generated labelled data derived from manufactured cases. In the next section, I propose using adjective antonyms as an evaluation standard for similarity based measures and argue that they are a suitable compromise between task-based and manual methods.

7.4.1 Adjectives and Antonyms

Semantic acquisition is a task that can be suitably represented by measuring or ranking word similarity. Thus an evaluation of *similarity measures* can be implemented using an objective list of similar words. A related approach was argued by Greffenstette (1993), who used available hand-coded resources such as thesauri and dictionaries to compare two different similarity measures. Greffenstette counted as correct pairs of nouns judged similar if they appeared in the same semantic category of a thesaurus (or in the case of dictionary definitions, had some degree of definition overlap). This strategy results in a coarse-grained approach at evaluation since the semantic categories are quite broad in the case of thesauri (averaging 60 words per category) and dictionary definitions (which often contain hypernymic relations making them increasingly general).

A more exacting standard can be obtained by using tighter lexical relations for comparison. Synonymy is an obvious choice but synonyms rarely appear in one-to-one relationships. For instance, WordNet (Miller, 1995) is organized around the concept of "synsets" which comprise sets of synonymous words, with no implied semantic similarity ranking within the sets.

The practical definition of similarity being measured in these experiments is the notion of "words that can appear in the same contexts", since the raw distributional information only contains contextual information. As I discussed in the previous chapter, distributional similarity can result from semantic relations besides synonymy. With this in mind, there is a strong case to be made for *antonyms* as a more precise distributional similarity standard. Antonymous adjectives are a unique class of lexical semantic relation. In contrast to most other relations, which form some type of *inclusive* semantic similarity (e.g. hyponymy, synonymy, meronymy), antonyms are semantically related *exclusively*. In addition, there appears to be a lexical component to the antonymic relation, manifest by the observation that close synonyms of antonym pairs do not yield the same strong associations (e.g. *big/little* vs. *large/little*). This makes antonymy an interesting case for the study of lexical semantic acquisition.

Justeson and Katz (1991) give a comprehensive and thorough treatment to the antonymic lexical semantic phenomenon. Their work is motivated by the assertion by

Charles and Miller (1989) that substitutability is *not* a suitable explanation for the strong antonymic associations reported in psycholinguistic testing. Charles and Miller argue that most adjectives occur in sentential contexts where an antonymic substitution would be improbable. Instead they suggest that antonymic associations are formed simply through frequent sentential co-occurrence. The main objective of Justeson and Katz is to provide empirical support for frequent antonymic co-occurrence, and to analyze the syntactic forms of these co-occurrences observed in the language.

In their experiments, Justeson and Katz collected frequency counts of antonymic co-occurrences for a set of antonym pairs compiled by Deese in 1964, which exhibited high correlations in word association tests, as well as a set of high-frequency antonyms and antonym pairs morphologically derived from negative affixes (e.g. a-, ab-, an-, dis-, il-, etc.). Using the 1,000,000 word Brown Corpus, they calculated the expected co-occurrence of an antonym pair using the mean of the hypergeometric distribution (i.e. the product of each antonym's individual frequency, divided by the total number of sentences). The number of antonymic co-occurrences actually observed in the corpus turned out to be significantly higher than the calculated expectation for a majority of the antonym pairs. Justeson and Katz conclude that this linguistic phenomenon might be a more plausible hypothesis than substitutability to explain the strong antonymic associations reported in psycholinguistic experiments.

Regardless of the psycholinguistic implications, the empirical evidence collected by Justeson and Katz suggests that antonyms possess properties that make them suitable as a standard for measuring distributional similarity. The higher-than-expected frequencies of co-occurrence reported by Justeson and Katz show antonyms will share many instances of the exact same context, which improves their distributional similarity making them more effective for evaluations of distributional similarity measures. Although Charles and Miller dispute the substitutability of antonyms, Justeson and Katz suggest that the co-occurrence of antonyms in the exact same context is something like substitutability, and therefore does not preclude antonyms from also appearing in independent similar contexts.

The arguments for similar distributions, together with the strong associations with their complements, help make antonym pairs less ambiguous than other lexical relations and a suitable choice for an objective measure of similarity. Although this approach still

employs a manual component in determining the antonym pairs, it is a simpler process than manually labelling data in a task-based evaluation. Furthermore, accepted lists of adjectives already exist like those employed by Justeson and Katz. Based on the evidence presented in this section, I will adopt adjective antonyms as a reference for evaluating the effectiveness of distributional similarity measures.

7.5 Experiments

In this section I present the results of implementing the three similarity measures described above and applying them to the task of ranking adjective antonyms. The distributional information was taken from the entire 100 million-word British National Corpus (2002) (BNC). Noun collocations for adjectives appearing in attributive form (e.g. *interesting film*) were collected and arranged into word frequency vectors. Collocations were determined using the part-of-speech tags supplied with the corpus, i.e. adjacent adjective and noun tags. This resulted in approximately five million events distributed over 90,000 adjectives.

Ten antonym pairs were chosen from the list compiled by Deese and referenced by Justeson and Katz (1991) in their work on antonym co-occurrence. Each adjective in the pair was required to appear with 25 or more unique noun collocations in the BNC. The selected antonym relationships used in evaluating the experiments are listed in Table 7.5.

Table 7.5
Antonym Pairs

light	*dark*	*big*	*little*
active	*passive*	*black*	*white*
alive	*dead*	*top*	*bottom*
back	*front*	*clean*	*dirty*
bad	*good*	*cold*	*hot*

Using the data in the BNC, distributional information was collected for each of the 20 adjectives listed in Table 7.5, resulting in word vectors comprised of noun event

frequencies. These raw vectors were then used in the various similarity experiments described below.

7.5.1 Scoring

A relative ranking of the adjective similarity was devised to score the similarity measures. For each adjective, the similarity measure under study was used to form a ranking of the other 19 adjective (vectors). The position of the true antonym in this ordered list was then used to calculate a reciprocal rank score. The reciprocal rank calculation was repeated for the other 19 adjectives and then averaged to obtain the Mean Reciprocal Rank (MRR) (cf. Voorhees and Tice, 2000). The MRR score ranges from 1/19 (true antonym always ranked last) to 1/1 (true antonym always ranked first).

To achieve a baseline for comparison, a random function was used to assign similarity scores for calculating a MRR. Twenty separate runs were averaged giving a random baseline (RB) score of 0.206. This score is not dependent on the configuration of the experiment since the random function did not use any of the distributional information.

The DC algorithm does not produce an absolute similarity score between word vectors as the other algorithms do. Instead, the DC algorithm produces distance scores between a word vector and the prototypical centroids of the derived clusters. This necessitated a heuristic for determining an absolute ranking given the derived clusters. Since every word vector is a member of every sense cluster, two word vectors can be approximately compared in relation to their distance to a cluster centroid. (This approximation becomes more accurate the closer the reference word vector is to the cluster centroid.) The heuristic involved finding the cluster whose centroid was closest to the reference word vector, and then using the absolute distances from the reference word to all other vectors in the cluster as the ranking metric.

7.5.2 Configuration

I measured the performance of the similarity measures for several different configurations created using three experimental variables. First, I used two sets of distributional data. A *small sample* comprising only the 25 most frequent events (i.e.

noun collocations) versus the *complete data* comprised of all events observed in the corpus. Second, I tried using the *root level classes* versus the *second level classes* of the WordNet taxonomy. Recall the WordNet classes are used to smooth the distribution of observed nouns into a practical number of classes. Each noun is replaced with its root or second level (depending on the experimental configuration) ancestor in the taxonomy. (See Section 7.3.1). Third, I used two configurations to deal with the polysemy of the events. In the first case only the primary sense information was included. This will be referred to as the *single sense* configuration. In the second case all sense information was included using the procedure described in Resnik (1993) (see Section 7.3.2 above). This will be referred to as the *multi sense* configuration.

The three configuration variables gave rise to eight distinct experimental configurations. For each of the three similarity measures under study, I ran all eight experimental configurations and recorded the MRR. The results are given in Table 7.5.2. The random baseline is also shown for comparison.

Table 7.5.2
Mean Reciprocal Rank (MRR) scores for three similarity measures and the Random Baseline (RB) on eight experimental configurations

Sample	Small Sample				Complete Data			
Class	Root Level		Second Level		Root Level		Second Level	
Sense	Single	Multi	Single	Multi	Single	Multi	Single	Multi
MMI	0.316	0.334	0.389	0.555	0.373	0.471	0.437	0.549
TAU	0.292	0.366	0.327	0.272	0.272	0.364	0.374	0.484
DC	0.233	0.333	0.299	0.309	0.346	0.411	0.309	0.342
RB	0.206							

Figure 7.5.3-1 - Single Sense vs. Multiple Sense

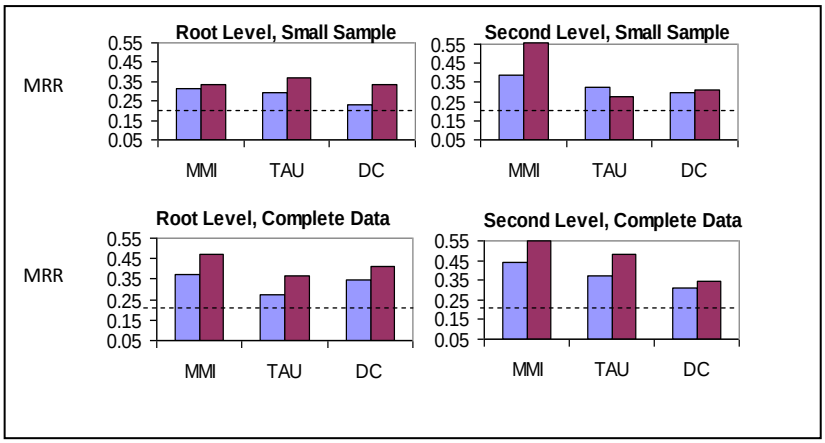


Figure 7.5.3-2 - Small Sample vs. Complete Data

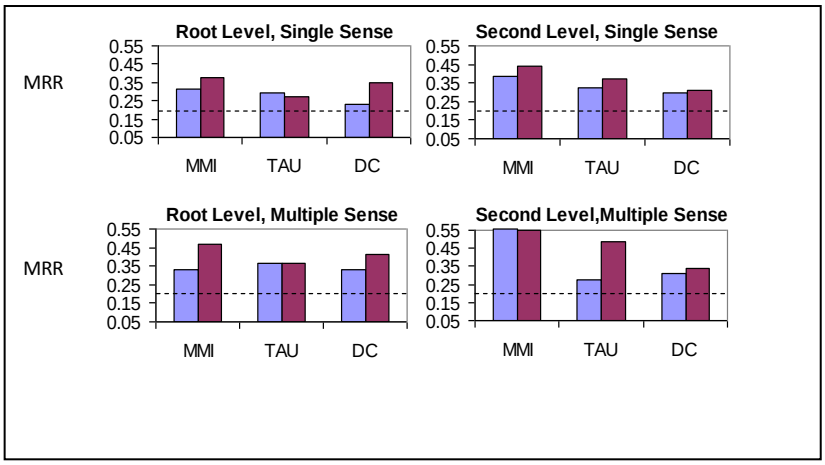
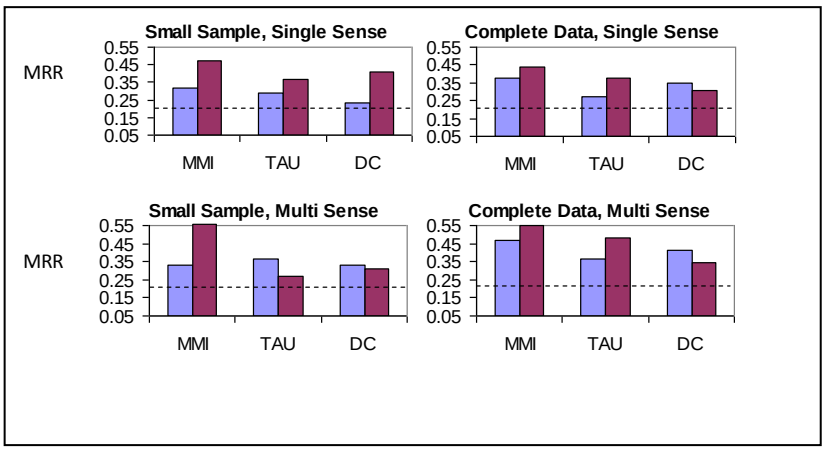


Figure 7.5.3-3 - Root Level vs. Second Level Classes



7.5.3 Results

All of the experimental measures performed better than the baseline, giving strong support to the hypothesis that antonyms produce similar distributions. The best performance was obtained using MMI, with the *small sample*, *second level classes*, and *multi sense* information.

The graphs in Figures 7.5.3-1, 7.5.3-2, and 7.5.3-3, show comparisons of performance with respect to each of the three configuration variables in the experiments: *single sense* vs. *multi sense, small sample* vs. *complete data*, and *root level* vs. *second level* classes. For each similarity measure in each graph two bars are shown. The first bar represents the results using the first configuration in the graph title (e.g. *single sense*) and the second bar represents the results using the second configuration (e.g. *multi sense*). The dotted line on each graph represents the random baseline score (which is invariant to the different configurations). In all cases the similarity measures performed well above the baseline, but these graphs reveal several other interesting trends.

1. Multi sense is better than single sense

As shown in Figure 7.5.3-1, using the nine *root level* concept classes, the MRR could be improved by including *multi sense* information. This was true regardless of the data sample size (i.e. *small sample* or *complete data*).

The same effect was observed using the 142 classes of the *second level* (see Figure 7.5.3-1), except in a single case. The TAU algorithm performs worse when *multi sense* information is included on the *small sample* data. The TAU algorithm is more sensitive to small fluctuations because it includes no magnitude information in its similarity calculation and disregards elements that are zero. With sparse class information over a large number of classes, this causes degradation in performance. To test this theory, I used a filter to remove very low magnitude elements from the vector (a value of 11 was experimentally determined). The results confirmed that after removing these noisy low-magnitude vectors, the *multi sense* configuration performed as well as the *single sense* data.

2. Using the complete data is better than using the small sample data

Figure 7.5.3-2 shows how the performance differs when using the *small sample* data versus the *complete data*. This also generally resulted in a performance improvement (or parity), with the exception of the TAU algorithm. A similar explanation as that used in (1) above can describe this decrease in performance. Using a filter as before to remove the (relatively) low magnitude elements improves the performance to .337. In this case the filter was determined to have a value of 330, which is small with respect to the magnitudes of the frequency information.

It should be noted that the best overall score was achieved using the *small sample data* with the MMI algorithm, although this is only marginally better (1%) than using the *complete data*.

3. Second level classes are better than root level classes for MMI and TAU

Figure 7.5.3-3 shows the performance delta when using the *root level* classes versus the *second level* classes. For the MMI algorithm this always resulted in better performance. This was true also for the TAU algorithm, except again in a single case: using *small sample* with *multi sense* distributions.

Examining the rank lists for this case reveals an anomaly. Most of the performance degradation can be attributed to the fact that for the adjective *cold*, the true antonym *hot* moves from being ranked first to fifteenth (i.e. a rank improvement of -14). This is atypical of the data when moving from the *root level* to the *second level* classes, as shown in Table 7.5.3-4. Although there are several other adjectives whose true antonyms slip in rank, none have the magnitude of *cold*.

Table 7.5.3-4
Rank Improvement for true antonyms using TAU when moving from root level to second level classes on small sample.

cold	-14	*alive*	-3	*top*	-1	*big*	3
clean	-6	*dirty*	-3	*bad*	0	*black*	3
hot	-6	*light*	-3	*passive*	2	*dead*	5
bottom	-5	*front*	-2	*back*	2	*active*	7
dark	-5	*good*	-1	*white*	2	*little*	10

Table 7.5.3-5
Statistics for the similarity scores produced by the TAU algorithm for the adjective *cold* with respect to all other adjectives

	Root Level Classes	Second Level Classes
Maximum	0.778	0.153
Average	0.424	0.105
Median	0.444	0.110
Standard Deviation	0.176	0.036

Comparing the similarity scores produced by the TAU algorithm for *cold* in both cases, we see very different statistics listed in Table 7.5.3-5. The maximum similarity score is quite high for the *root level* (0.7778), which corresponds to the true antonym *hot*. For the *second level*, the maximum similarity score is 0.153, and corresponds to the (wrong) adjective *clean*. Although this appears to be a very low score, this is only moderately low in comparison to other scores on the second level. However, the average, median, and standard deviation show that the similarity scores produced by the TAU algorithm for *cold* in this case occupy a very narrow range. There are no strong similarities for *cold* when using the *second level* classes with only the *small sample* data. The distribution for the adjective *cold* in this case does not give a distinct profile when dispersed over the 142 *second level* classes. The distinctiveness (and similarity to the true antonym) re-emerges when using the *complete data*, improving the rank considerably.

4. DC performs worse using second level versus root level classes.

For DC, moving from the *root level* to the *second level* classes degrades performance in three out of four cases. The DC algorithm measures similarity using all available dimensions. The larger dimensional space of the *second level* classes gives much more freedom for the cluster centroids of the algorithm to associate with the adjectives. This introduces more opportunities for spurious similarities which can eclipse more genuine

relationships. In fact, at the *second level*, DC performed worse than the other algorithms in all cases but one (*small sample, multi sense*).

5. MMI is the best algorithm to use

Overall the MMI algorithm performed the best, and was the most robust to the various configurations. The MMI algorithm also followed the intuitive expectation that similarity measures should improve with polysemy information (*multi sense*), more distributional data (*complete data*), and finer grained classes (*second level*). The basis for the MMI algorithm is the mutual information measure of two linguistic entities, in this case two adjective word vectors. There is strong evidence that the hypothesis of similar contexts for antonymic adjectives is correct, or at least helpful, since using this algorithm results in a similarity measure which performs much better than chance.

Another further advantage of the MMI algorithm is the computational complexity. Among the three similarity measures investigated, the MMI algorithm requires the least amount of computation. DC is a highly computationally intensive algorithm involving simulated annealing, which requires constant perturbation and re-estimation of the cluster centroids. The TAU algorithm requires calculating the differential of every pair of vector elements, which increases non-linearly in the number of elements. The MMI algorithm only requires a linear processing of each vector to produce the elemental probabilities.

One disadvantage of the MMI algorithm is the requirement to have complete data for all vectors before the similarities can be computed. Each elemental probability calculation requires the total number of all events, and the total number of the elemental events. The TAU algorithm does not have this constraint, since it only uses elemental differentials. After calculating a similarity measure using the TAU algorithm, subsequent unrelated events would not affect the differentials already calculated. For the MMI algorithm, however, this would require a complete recalculation of all scores.

7.6 The antonym pair *good*/*bad*

Among the antonym pairs used in the experiments, *good*/*bad* were consistently ranked most similar to each other by all the algorithms. Table 7.6 shows the average

similarity rank (over all algorithms and all configurations) of an adjective by its true antonym versus the average similarity rank by all other adjectives. For the pair *good*/*bad* we see that on average each ranks the other as highly similar (1.67/1.38). In contrast, all other adjectives tend to have *good* and *bad* very low in their similarity rankings on average (13.98/13.39). So it is not the case that *good/bad* simply have distributions that cause them to be ranked highly similar to any word by default, as for example *clean/dirty* which have very similar average rankings by both themselves and the other adjectives (6.71 vs. 8.39, 8.17 vs. 9.48).

The performance of *good*/*bad* suggests their distributional profiles are distinctive enough, compared with the distributions of other adjectives used in the study, that the similarity measures are able to discriminate them more easily. One possible explanation is that the antonyms *good*/*bad* have a much wider distribution over all semantic categories of nouns than the other antonyms, owing to their generality. This would produce a much "flatter" distributional pattern that is a poor match for the more specialized distributional patterns of the other adjectives.

Table 7.6
Average Similarity Rank of an Adjective by its Antonym and Other Adjectives over all configurations

	Antonym	Others		Antonym	Others
good	1.67	13.39	*black*	2.92	9.04
bad	1.38	13.98	*white*	3.67	8.48
alive	7.08	12.72	*top*	7.17	8.78
dead	13.04	8.90	*bottom*	6.17	9.96
back	6.33	10.92	*clean*	8.17	8.39
front	8.17	9.56	*dirty*	6.71	9.48
active	4.92	12.28	*cold*	4.25	10.31
passive	5.33	12.25	*hot*	6.21	9.10
big	7.21	10.13	*dark*	10.5	7.08
little	5.75	10.35	*light*	7.92	8.93

7.7 Conclusion

In this section, I have attempted to show that similarity measures based on distributional information can be successfully applied to tasks having objective evaluations (e.g. semantic acquisition). Using adjective antonyms as the evaluation metric I was able to objectively characterize the performance of three different similarity measures. Regardless of the algorithm or the experimental configuration,

using the distributional information to measure adjective similarity successfully ranked the true antonym higher than others, with greater frequency than by chance.

Although antonyms are not normally considered to be similar, they confer similar properties on the head nouns they modify through attribution. As a result, antonyms have very similar distributions making them uniquely suited for a task-based evaluation of distributional similarity measures.

The experimental results show that of the three similarity measures tested, MMI gives the best performance on the widest range of configurations. It is less computationally complex, and has a strong theoretical motivation. MMI achieved the highest score when polysemy was considered in the distributional information of the events and a larger number (i.e. finer distinction) of the conceptual classes was used for the elements of the adjective word vector. However, MMI requires complete data for all vectors before calculating similarity (see Section 7.5.3) which may make it unsuitable for some applications. In this respect, the TAU similarity measure may be an acceptable compromise between performance and feasibility.

8

Characteristic Adjectives

In the previous chapter I examined the properties of statistically based similarity measures by evaluating several distributional approaches to semantic class acquisition of adjectives. In this chapter I will continue exploring statistical methods of language processing, using adjectives to discriminate nouns. To address the issues encountered when applying the vectors derived from large corpus processing, I will develop the idea of *characteristic adjectives* as a filtered set of adjectives that highly correlate with a (nominal) node in a semantic taxonomy (e.g. WordNet), and discuss various approaches to determining these characteristic adjectives, and their limitations. I will then propose one approach that can be used to derive them successfully. Finally, I will present the results of experiments designed to test the hypothesis that characteristic adjectives can accurately predict a semantic node.

8.1 Characteristic Adjectives

One drawback to using the three semantic similarity measures evaluated in the previous chapter is the computational cost. Sophisticated statistical measures such as DC are expensive in terms of time and complexity. Measures such as MMI require matrices of all vectors and features discovered in a corpus to compute similarity, which can sometimes be impractical. It would be preferable to find a simpler and more efficient measure of similarity that is comparably effective.

Another disadvantage to using the previous measures is they rely solely on statistical information which limits their use to certain types of corpora. Statistical approaches are appropriate for finding the similarities between words that appear in a large corpus, where each word has a large number of occurrences. However, if the similarity is to be calculated for words that only occur a relatively small number of times in a corpus,

statistical approaches may be less effective, i.e. the smaller the number of events the larger the potential statistical error.

For example, the vectors derived from a large corpus could be used to annotate the class information in the CAMEO representation to augment a language task as described in Chapter 6. When the task processes a document (comprising a different corpus), it may encounter unknown words and attempt to measure distributional similarity against the annotated class vectors. To do this, the task would need to derive a distributional vector for the unknown word in the context of the document. However, the distributional vectors that can be derived from the document will be a fraction of the size of the vectors annotated from the large corpus. Consider the vectors of adjectives derived from the BNC that were used in the experiments in Chapter 7. The size of these vectors ranged from several hundreds of unique tokens to a thousand or more. The same procedure used on a single document will normally produce vectors having on the order of tens of unique tokens. Statistical similarity measures like those described in Chapter 7 use correlations between vector elements to calculate distance, and the large number of elements in the BNC vectors makes it likely that they will all have spurious correlations with the much smaller vectors in the document. Thus the similarity measures will be less effective at distinguishing which BNC vector is most semantically related to a word in the document.

One possible solution to this problem is to try to determine which elements of a distributionally derived vector are the most salient. In other words, find which elements carry the most semantic information about the vector at large. For instance, using the same methods as described in Chapter 7, vectors of adjective collocations can be derived for nouns. For these vectors the goal would be to determine which of those adjectives carry the most semantic information about the noun. Retaining only the most informative adjectives would result in smaller vectors that could then be used to measure semantic similarity of nouns in the context of a smaller corpus (e.g. a small set of documents). Further, these smaller vectors should only require a very simple similarity calculation because they have already been determined to be semantically significant.

A small set of adjectives like this which are semantically significant for a given noun can be said to be *characteristic* of that noun, because they attribute characteristic

properties. Characteristic adjectives then, are a small set of adjectives that indicate a high semantic correlation with a noun when observed in the corpus. I will test this hypothesis in the experiments that follow, but first I will investigate ways to discover sets of characteristic adjectives.

There are several existing techniques that can be applied to reduce the dimensionality of a vector. I used class-based smoothing in the experiments in Chapter 7 to conflate multiple individual word tokens into fewer, more general semantic classes. However, class-based smoothing requires broad semantic categories, which are less understood for adjectives. Other common approaches to vector reduction are Principle Component Analysis (PCA) and Singular Value Decomposition (SVD), which are statistical data-driven approaches to finding a smaller set of salient vector elements, typically used in IR algorithms such as Latent Semantic Analysis (LSA). Data-driven approaches like PCA and SVD are general dimensionality reduction techniques not directly based on the intrinsic properties of the vector components (although their application to natural language is motivated by the distributional hypothesis of similar words). By contrast, the approach I will develop leverages the inherent properties of adjectives, and the related semantic hierarchy of the nouns they modify, to find semantically salient components irrespective of the statistical data.

In order to determine the best approach to deriving characteristic adjectives, it is helpful to look at the lexical properties of the nouns they describe. A lexical taxonomy, such as WordNet, organizes nouns in a conceptual hierarchy (e.g. hypernymy/hyponymy). The hyponymy relationship is a specialization function (e.g. *dog* is a specialized form of the more general *animal*), and hyponyms share all properties of their ancestors. Therefore, adjectives that can modify a certain noun can also appear with all of that noun's descendent nodes[2]. For instance, *hungry* can modify *animal* and any of its hyponyms (*hungry dog*).

[2] Note that the WordNet taxonomy includes information relating adjectives to nouns. For descriptive adjectives, which specify the value of a nominal attribute, WordNet points to the synset representing the attribute. For example, the adjective *dry* contains a pointer to the noun (synset) *wetness*. Relational adjectives, which are derived primarily from nouns, have pointers to the related noun (synset), e.g. the adjective *idyllic* contains a pointer to the

A characteristic adjective should represent a unique property of a noun. For this reason, we cannot simply take the most frequently occurring adjectives as a noun's characteristic adjectives. An adjective may actually be more characteristic of an ancestor node much higher in the semantic hierarchy. Instead, it is necessary to distinguish when an adjective is characteristic of a given node, and when it is simply inherited.

8.2 A Bottom-Up Approach

One approach to deriving characteristic adjectives is to use a bottom-up strategy. This strategy has limitations (as I will show), but it is a useful first approximation of an algorithm that can later be refined. In the next few sections I will explore the bottom-up approach and its limitations, and use it as basis for comparison to develop an alternative approach.

In the bottom-up approach, distributional information is used to populate a conceptual taxonomy with collocative adjectives, which are then post-processed by recursively 'percolating' them up the hierarchy. Adjectives and their corresponding attributes are ultimately based on physical properties of the objects they modify (e.g. *long* and *short* imply an object with some measure of *length*). These properties will be implicit in the semantic organization of the conceptual hierarchy, and the adjective distributions should reveal this.

As I previously noted, an adjective observed at a given node in the tree is not necessarily characteristic at that node. The properties of a parent node exist in all

---

noun (synset) *idyll*. For the experiments in this chapter, the distributional properties of adjectives are considered, which implicitly derive from these relations. Although it may be possible to use this information explicitly to smooth adjective events into attribute or relational classes, this aspect was not explored in the current work.

descendant nodes, but they are only characteristic in the context of the parent. For instance, the adjective *living* may help distinguish an *organism* from an *artifact*, but it is useless when distinguishing between two organisms (*cat* vs. *dog*). For the bottom up approach to work, some decision algorithm must exist to decide when an adjective needs to be "pushed back" up the semantic chain. That is, there must be a means of deciding when an adjective is characteristic and when it simply indicates an inherited property. If we observe the same adjective for all (or most) children of a parent node in the semantic hierarchy, we might assume it indicates a property inherited through the parent node. By processing the semantic tree from most specific to most general (i.e. bottom up), observed adjectives can be pushed recursively up the tree to the highest node that does not share the adjective with its siblings. This then becomes a characteristic adjective, distinguishing the node in the context of its siblings.

As an example, consider the deverbal adjective *married*. We might expect, using the algorithm described above, for this adjective to eventually be assigned as characteristic of *person*. The adjective *married* depends upon, and subsequently attributes, the properties of being a person. We may observe instances of the sibling nodes *married man* and *married woman* and determine that we are justified in assigning this adjective to the parent node *person*.[3]

The bottom up approach suffers from serious limitations, which I explain in the following sub-sections. In section 8.3 and 8.4 I will describe how to avoid these limitations by using a slightly different approach.

8.2.1 Data Sparseness

There is a data sparseness issue that arises when using the bottom up approach. The problem, however, is not that there are not enough adjectives observed for a given noun, but rather the distribution of these adjectives is sometimes insufficient for a semantic node.

---

[3] It may be argued that *person* is too broad a category for which to assign *married*, since e.g. *baby* and *pope* are both children of the *person* node. The taxonomy may not include a node representing the precise set of characteristics, e.g. *marriable person*. However, because characteristic adjectives are not used in a generative capacity, having a wider scope is not an issue.

Recall that the decision to promote an adjective to a parent node depends on the adjective being observed with all its children. In practice this rule does not work because not all children can be expected to co-occur with an adjective. There may be some children of a node that are quite rare and do not even occur in the corpus. This is often true in WordNet, which strives to be so comprehensive as to include esoteric terms, euphemisms and slang among its entries. However, even for more common terms which do appear in the corpus, an adjective describing a legitimate property may be unlikely to co-occur. Returning to the *married person* example, WordNet has over 300 hyponyms for the synset headed by the concept *person*. For most of these it would be uncommon to find them modified by the adjective *married*. For example, *married waker*, *married captor*, and *married nonworker*, although semantically acceptable, are somewhat unexpected. Contexts where these might appear would be unusual and most likely involve some distinguishing discourse level context.

This data sparseness skews the distribution of *married* over the children of the *person* node, making it difficult to derive a general rule for promoting an adjective. It may be possible to adjust the algorithm heuristically to use some soft threshold based on, say, a weighted percentage of the observed instances over all the child nodes, but ultimately any empirically determined parameter will be incapable of correctly handling all possible configurations of nodes.

8.2.2 Polymorphism

The other major obstacle to using a bottom up approach for deriving characteristic adjectives is polymorphism. In natural language, it is allowable for a more general concept to be substituted for a more specific concept. This is known as polymorphism because the general term is able to change and assume the properties of the more specific term. For example, it is semantically acceptable to substitute *poor thing* for *poor person*. In this case *thing* is standing temporarily for the *person* class and can no longer be said to be a member of the *thing* class.

This behaviour is problematic for distributional techniques because the events observed with the polymorphic object do not necessarily belong to it. In the above example, the observed adjective *poor* should definitely not be associated with *thing*,

since that would ascribe the implied attributes of *poor* to all descendants of the node *thing* (e.g. *molecule*, *ocean*, etc.).

There is no simple way to determine when an object is functioning in a polymorphic manner. Without a means to determine this, adjectives can appear almost anywhere on the conceptual tree making the derivation of characteristic adjectives very difficult.

8.3 Characteristic Attributes as Differentiae

Polymorphism and data sparseness expose the weakness in the bottom up approach to deriving characteristic adjectives. The bottom up method relies on aggregating information recursively from the bottom of the tree and so is vulnerable to incomplete information. A better approach is to use the differential of a node's observed adjectives compared with those of other proximate (i.e. child) nodes. So rather than aggregating and promoting the adjectives that appear in a majority of child nodes, this approach would eliminate adjectives appearing in child nodes, and retain only those adjectives that appear as unique.

This approach is suggested by the organization of the semantic taxonomy, which parallels to some degree the organization of plant and animal taxonomies found in the biological sciences. A parent node in the semantic hierarchy can be thought of as the biological genus, and child nodes to species. A genus is defined as a group where each member has a significant number of shared characteristics. A species is a member of this group that can be distinguished from other members by one or a small number of characteristics. Thus in the differential approach, we are explicitly deriving the distinguishing characteristics of a species node. One way to find this set of characteristic adjectives would be to simply compare a node's vector with all other vectors. Any adjectives that only appear with a single node could be taken as characteristic. Of course, in practice it would not be feasible to compare a node's adjectives with that of all other nodes. In fact this would not necessarily be desirable since homonymy and polysemy will produce legitimate multiple occurrences of the same adjective on nodes that do not coincide semantically (e.g. *a large (striped) bass*, *a large (string) bass*).

Instead, it should only be necessary to look at a local portion of the tree when computing the differential for a node. This local area will have a tighter semantic correlation and adjectives that occur multiply within this context will likely be implying the same attributes (and can thus be ruled out as characteristic).

Note that a characteristic adjective does not necessarily occur with the node it belongs to. Polymorphism and inherited properties make it possible for the adjective to occur anywhere in the node's set of descendents. In fact the bottom up approach was based on the idea that it would be necessary to "push" these adjectives up to their proper nodes. Therefore, when using differentials to determine characteristic adjectives, it may be necessary to aggregate the vectors of descendent nodes with the vector of the parent node under investigation.

Theoretically, it would be possible to include the entire tree of a node's descendents when determining characteristic adjectives, but in practice this approach would be unfeasible for nodes anywhere near the root. The set of adjectives would become too large, and the effects of polysemy and homonymy could appear because of the semantic scope included in such a large portion of the tree. Using a much smaller sample should not hinder the results significantly, although it is possible that some characteristic adjectives would be missed in this case (namely characteristic adjectives that only appear with descendent nodes beyond the restricted portion of the tree). As I demonstrate in my experimental results however, this did not appear to be a significant issue.

8.4 Experiments

The experiments presented in this section are designed to test the hypothesis that characteristic adjectives can be used as a measure of semantic similarity. For a semantic node in a taxonomy (e.g. a synset in WordNet), a vector of characteristic adjectives is hypothesized to have less statistical noise than the complete vector of adjective co-occurrences (with respect to a particular corpus). In other words, reducing the vector representing a node to its most salient adjectives gives it sharper semantic focus. Thus a vector of characteristic adjectives should possess a better capacity for discriminating vectors of similar semantic terms.

There are two major parts of the experiments to consider described in turn below; the method for determining the vectors of characteristic adjectives, and the method for testing the measure of similarity. The data for the experiments was taken from the BNC. Using methods similar to those described in Chapter 7, vectors of adjective collocations were derived for nouns in the corpus. The semantic taxonomy used in the experiments was WordNet 2.0. WordNet contains nine "unique beginner" nodes which serve effectively as root nodes for separate taxonomies. Within each of these taxonomies, all nodes on the first two levels were tested.

```
Start
    For each unique beginner node nu
        Call procedure Determine characteristic adjectives with nu
End

Procedure Determine characteristic adjectives uses node n
    For each child c of node n
        Call procedure sum vectors with c
        Remember vector of c
    For each child c of node n
        For each sibling s of c
                For each adjective a in vector of s also appearing in vector of c
                Remove a from vector of c
        Save vector of c
End

Procedure sum vectors uses node n
    For each noun w in synset of node n
        Sum vector of w with vector of n
    For each child c of node n
        Sum vectors of all nouns in synset of c with vector of n
    Return vector
End
```

Figure 8.4 –Pseudocode for deriving characteristic adjectives

Figure 8.4 gives pseudocode for the derivation of the characteristic adjective vectors used in the experiments, according to the approach described in Section 8.3. Each noun is represented by a vector of adjective events observed in the corpus. The vectors are manipulated (either through aggregation of several vectors or removing adjective elements from individual vectors) to derive vectors of characteristic adjectives. For example, a node in the taxonomy represents a synset having one or more noun synonyms. To derive the vector for a node requires combining the vectors of all individual nouns in the synset, as shown in Figure 8.4.

For each node to be tested, I combined the adjective vectors of all nouns in its synset, along with all nouns in all synsets of its immediate children. The same procedure was performed on all of the test node's siblings. Using this set of vectors, the differentials were computed by comparing the test node's vector to its siblings' vectors and removing any adjectives that appeared in common. This resulted in a vector composed of all the (locally) unique adjectives which were observed in the corpus appearing with one of the nouns in the test node's synset (or immediate descendent synsets).

The derived characteristic adjective vectors were then used to select nouns from the BNC corpus by determining a vector-based similarity score. Because the characteristic adjective vectors are hypothesized to be composed of unique differentiators, a sophisticated similarity measure should not be necessary. Additionally, using a sophisticated similarity measure would conflate the performance of the characteristic adjectives with the performance of the similarity measure itself. Instead, a direct matching similarity metric should give a better indication of the associative strength of the characteristic adjectives. For these reasons a simple similarity score was used consisting of the the number of matching adjectives as a percentage of the total number of adjectives in a noun's vector.

For each node under test, I scanned all nouns in the BNC corpus, matching a noun's vector of adjectives against the test node's derived characteristic adjective vector. The similarity score was computed as the number of matching adjectives as a percentage of the total number of adjectives in a noun's vector. The highest similarity scores select the most similar nouns to the node under test. (A 25% similarity score threshold was determined empirically and simply gives a static level on which to base comparison). I used both the derived characteristic adjective vector and the complete adjective vectors to perform the experiments and compare the results.

8.4.1 Evaluation

Characteristic adjectives can be seen as a filter, selecting for the semantic node (synset) from which they were derived. To judge their effectiveness we can measure how many nouns (i.e. synonyms) are selected belonging to the synset, in proportion to

the total number of nouns that are selected. This gives us a basis for calculating precision *P* and recall *R*.

P = synonyms selected / all nouns selected

R = synonyms selected / total synonyms possible

Another measure of the effectiveness of characteristic adjectives is looking at the quality of the nouns that are selected. We expect the nouns selected, if they are not members of the synset, to at least be close semantic relatives of the node. More precisely, we would expect them to be descendents because the set of characteristic adjectives define distinguishing characteristics of the original semantic node. All nouns found having these characteristics should inherit them from this node and therefore be descendents.

We can measure the quality of the selected nouns by computing their distance to the semantic node. Call this distance the *span*. The span is computed by counting the distance (edges) between the node of a selected noun and the semantic node under test. If the selected noun is not a direct descendent of the node, the nearest common ancestor is used. Figure 8.4.1 shows an example of measuring the span between a selected noun *nib* and a test node *tip.* The selected noun *nib* is related to the test node *tip* only through the common ancestor *end*.

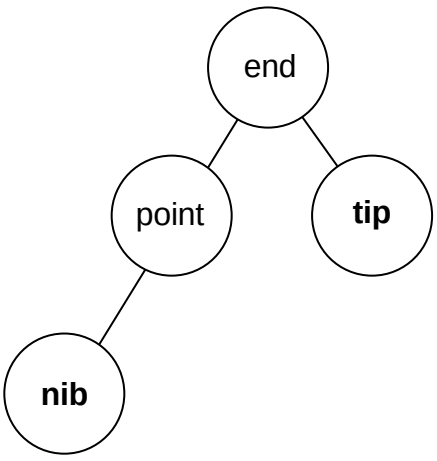


Figure 8.4.1 –Example of measuring the span between *nib* and *tip*

Note that nodes closer to the root of the tree are increasingly general and will include larger portions of the tree. Thus the likelihood of two nodes being related increases for nodes closer to the root. To adjust the span score for this root proximity effect, a factor is included relative to the distance from the root of the common ancestor node. This gives less weight to the score when the nearest common ancestor is close to the root of the tree, which usually indicates the selected noun is only distantly related.

The formula for the span *s* is calculated as

$$s = (d_{root} + 1) + \frac{1}{1 + d_{to}} + \frac{1}{1 + d_{fro}}$$

where d_{root} is the number of edges between the common ancestor node and the root node, d_{to} is the number of edges from the selected node *to* the common ancestor, and d_{from} is the number of edges *from* the common ancestor to the test node.

The span metric presented here is similar to other path-based measures (e.g. Leacock and Chodorow, 1998), especially Wu and Palmer (1998). Wu and Palmer also incorporate the distance to the root from the nearest common ancestor of two nodes being measured in their similarity score. However, they scale this distance by a factor of 2, giving it much more weight than the distances d_{to} and d_{from}, compared with the span calculation proposed for these experiments. Wu and Palmer also use a form that is normalized between 0 and 1, which I chose not to do since the absolute values give information about a test node's placement in the semantic hierarchy. The span in this case is only used as a relative comparison between specific levels in the hierarchy and not as an absolute measure (which would require normalization).

For the example given in Figure 8.4.1 above, the distance from the common ancestor *end* to the root node *entity* is 4 edges (*end →extremity→region,part→location→entity*). The distance from the selected node ***nib*** to the ancestor *end* is 3 edges, and the distance from *end* to the test node ***tip*** is 1, giving a span score of

$$s = (4 + 1) + \frac{1}{1 + 3} + \frac{1}{1 + 1} = 5.75$$

Note the calculated span has a minimum value based on the distance of the test node to the root. In the best case, if a selected noun is a synonym of the test node, both d_{to} and d_{from} will be 0 and the nearest common ancestor will be the test node itself, giving $s_{\max} = d_{root} + 3$. Also, because the span score is not normalized, span scores are only useful for comparing the quality of nouns selected by vectors from sibling nodes (which reside at the same level of the tree).

Each selected noun has the possibility of being polysemous, which poses a difficulty for the evaluation. But since a noun is being selected using the vector of characteristic adjectives (which are semantically related to a single sense of the test node), it is fair to assume the most appropriate sense for the selected noun. Thus, for the purposes of the evaluation, the distance to the test node was computed for each sense of a selected noun, and the sense in closest proximity to the test node (i.e. the best match) was used to calculate the span score.

The baseline used in the experiments consisted of the complete (undifferentiated) vectors of adjectives. That is, the vectors before removing adjectives found in common with sibling nodes. Using these undifferentiated adjectives provides a baseline measure for the precision, recall, and span scores of a test node. We would expect the vectors used in the baseline to be less discerning and select a wider semantic scope of nouns because the vector of adjectives would be more general. We can predict that the baseline will have a greater recall with a lower precision. We would also expect the baseline to have higher (worse) span scores because of the wider semantic breadth the undifferentiated vector would cover.

8.5 Results

I calculated characteristic adjective vectors for the first three levels of the WordNet taxonomy, comprising 720 nodes. Using these differentiated characteristic adjective vectors, along with the undifferentiated baseline vectors, I scanned all nouns in the BNC corpus. Each vector selected a set of nouns matching the threshold percentage of the vector's adjectives (described above).

For each vector I calculated the average precision *P*, recall *R*, and span *s* over all selected nouns. I then averaged these scores over sibling nodes and assigned the scores recursively up to the nine unique beginner nodes. The statistics for the nine unique beginners are shown in Table 8.5.

For each root node, values are given for the averages derived from the characteristic vectors, the baseline vectors, and the differences between the two (Delta). Averages are reported as the number of true synonyms selected by the vector (Synonyms), the total number of nouns selected (Total Selected), the average span of all selected nouns (Avg Span), and the calculated Precision and Recall.

Table 8.5
Results for selecting nouns using the nine unique beginners

	Characteristic Vectors					Baseline Vectors					Delta				
Root Node	**Synonyms**	**Total Selected**	**Avg Span**	**Precision**	**Recall**	**Synonyms**	**Total Selected**	**Avg Span**	**Precision**	**Recall**	**Synonyms**	**Total Selected**	**Avg Span**	**Precision**	**Recall**
Entity	16	951	127.85	.017	.727	20	25112	178.93	.001	.909	-4	-24161	-51.080	.016	-.182
Act	43	55745	213.79	.007	.417	54	47866	199.10	.001	.621	-21	-42121	14.690	.006	-.204
Abstraction	6	587	62.74	.010	.316	8	14836	86.72	.001	.421	-2	-14249	-23.980	.010	-.105
Event	10	500	39.10	.020	.556	13	14551	22.32	.001	.722	-3	-14051	16.780	.019	-.167
Psych	12	198	72.03	.061	.414	12	16725	86.78	.001	.414	0	-16527	-14.750	.060	.000
Phenomenon	7	1415	35.35	.005	.412	10	12282	18.09	.001	.588	-3	-10867	17.260	.004	-.176
Group	22	2103	127.48	.010	.423	23	35720	98.90	.001	.442	-1	-33617	28.580	.010	-.019
Possession	4	520	24.02	.008	.444	4	9214	8.74	.000	.444	0	-8694	15.280	.007	.000
State	49	12624	206.28	.004	.333	60	75548	174.79	.001	.408	-11	-62924	31.490	.003	-.075
Total	169	24643	908.64	.007	.406	214	251854	874.37	.001	.514	-45	-227211	34.270	.006	-.108

8.5.1 Quantitative Analysis

The aggregate statistics show that for each of the nine unique beginners, using characteristic adjectives improved precision while worsening recall in a much smaller proportion. For instance, the characteristic adjective vector for the root node *entity* selected four fewer synonyms than the baseline vector, but the total number of nouns selected was 24,161 less. Thus the recall worsened by 20.1%, but the precision improved by 1,700% (.017 / .001). The same pattern is observed when viewing the

aggregate of the first level nodes under a unique beginner, and continues to the individual nodes themselves.

Out of 720 individual nodes under study, there were 149 cases where using characteristic adjectives improved the precision over the baseline. The average improvement in precision over all individual nodes was 10%, but there are many examples of improvements of 90% or more. There were 35 cases where the precision worsened, but the average degradation was so small as to be almost undetectable. These cases are typified by instances where the baseline finds a single synonym but has a huge number of false positives giving it a very small precision. The characteristic set restricts the false positives but if it loses the synonym in the process the precision goes to zero.

The experiment shows there is a significant reduction in ambiguity using the much more restrictive characteristic adjectives. In most cases the number of nouns selected dropped by 90-95% of the numbers using the baseline. The fact that the recall only worsened in 40 cases seems to indicate the characteristic set is describing real features of the noun class.

Although using true synonyms in scoring provides a clear measure of precision and recall, there are also legitimate hyponyms that could be considered as properly selected. The average span (described in Section 8.4.1) is used to measure this. There were 249 cases where the average span improved and 442 cases where it worsened. Of the 442 worse cases, 380 of these were due to null membership for the characteristic adjectives. That is, cases where the characteristic adjectives did not select for any nouns in the corpus. This happens when the adjective set is so small to begin with, that differentiating leaves too few, if any, characteristic adjectives.

8.5.2 Qualitative Analysis

It is informative to look at the kinds of adjectives that comprise a characteristic set after the differentiation process. Table 8.5.2-1 lists the top 20 characteristic adjectives (based on frequency) for *person*, which is the first child of the *cause* node under the *entity* root. Table 8.5.2-2 lists the top 20 adjectives in the original baseline vector (before differentiation). Adjectives in bold appear in both sets.

There were 1,367 adjectives in the baseline vector, and 794 characteristic adjectives after differentiation, giving a ratio of roughly over half the adjectives retained as characteristic.

Comparing the two tables it is clear that many of the adjectives in the baseline vector do not imply attributes that are unique to a *person*. Adjectives such as *old*, *only*, *other*, *ordinary*, and *peculiar*, which appear in the baseline vector and not in the characteristic vector, either imply attributes that are universal (*old*, *only*) or are deterministic (*other*, *ordinary*, *peculiar*). Contrast these adjectives with those found in the characteristic set such as *bereaved*, *unemployed*, *insured*, and *immortal*, which all imply distinctly human attributes (within the local semantic context of the *person* node).

Table 8.5.2-1
Top 20 characteristic adjectives of *person*

Word	Count	Word	Count
bereaved	31	unemployed	24
vulnerable	22	deaf	52
deceased	24	**elderly**	189
accused	17	healthy	26
ill	16	infected	27
insured	57	lay	17
living	41	missing	15
named	32	qualified	28
assisted	14	sensible	22
sick	25	immortal	22

Table 8.5.2-2
Top 20 baseline adjectives of *person*

Word	Count	Word	Count
young	262	dead	71
different	68	disabled	107
elderly	189	human	53
important	55	**insured**	57
old	189	older	80
only	428	ordinary	65
other	743	particular	129
poor	70	private	79
real	66	right	77
authorised	59	single	245

There are some characteristic adjectives that do appear not to imply uniquely human attributes such as *missing*, *healthy*, *sick* and *living*. However, these adjectives are characteristic in the context of the parent (genus) node, and serve to distinguish the child (species) node from its siblings. In this context there needs only to be a differentiation from other children of the *cause* node. Other children of the *cause* node include:

agent:: a substance that exerts some force or effect

supernatural, occult:: supernatural forces and events and beings collectively

destiny, fate: the ultimate agency that predetermines the course of events

engine: something used to achieve a purpose

The full baseline vector for the *person* node contains 1,367 adjectives and selects 23 nouns from the BNC corpus. The characteristic adjective vector contains 794 adjectives and selects only 5 nouns. These two lists are shown in Table 8.5.2-3. Although there are several nouns that should be selected by the characteristic set (such as *child* and *women*), most of the nouns in the baseline list are spurious (*part*, *place*, *state*, *thing*) and the characteristic set successfully filters these out, while retaining those nouns that are semantically close to *person*.

Table 8.5.2-3
Nouns selected by vectors of node *person*

characteristic set	baseline set		
man	body	men	state
men	business	nature	thing
people	character	part	way
person	child	people	woman
woman	family	**person**	women
	form	place	work
	group	sense	world

8.6 Conclusion

In this chapter I have attempted to show that a distributionally derived vector of differentiated adjectives can be used to represent a semantic node in a taxonomy. These *characteristic adjectives* are motivated by the attributive lexical function of adjectives, and are another example of the types of statistical processing techniques that can be applied to language processing. Characteristic adjectives give an alternative similarity measure for class based semantic acquisition, compared with those examined in the previous chapter. Rather than using a more complex similarity calculation over

comprehensive distributional data, this method attempts to find a smaller number of distinguishing data, which can then be used with a simpler similarity calculation.

I looked at several approaches to deriving characteristic adjectives, motivated by insights into the properties of adjectives. Of these approaches, the most successful was using differentials of neighbouring semantic nodes. By removing those adjectives that appear in the vectors of nearby semantic nodes, the distributional data is reduced to the most distinguishing features.

The experiments appear to confirm the semantic properties of the characteristic adjectives and their ability to determine semantic similarity. Qualitatively, the types of adjectives that are derived as characteristic appear plausible as representing distinguishing attributes, which is one motivation for their development. Quantitatively, the characteristic adjective vectors improve the correlation to the correct semantic node, in comparison to the complete distributional vectors.

The derivation of characteristic adjectives presented in this section can be affected by polysemy, as with most distributional methods. However, because in this case the highest level semantic nodes are used, the nouns under study are very general and mitigate these effects to some degree. Polysemy is likely to be more pronounced for nodes lower in the taxonomy and may degrade the performance of differential metrics such as characteristic adjectives. One possible way to address this would be to use the distributions in the vicinity of homonymous nouns to filter adjectives in the polysemous distribution, but this would complicate the algorithm and warrants further investigation.

Another goal of this chapter has been to explore some of the possible approaches enabled by corpus based distributional methods. Results from these types of experiments have the potential of aiding symbolic language tasks by integration with a framework such as the CAMEO representation. For example, the distributionally derived characteristic adjectives could be used to augment the symbolic representation of a document (or set of documents) processed in the system using the annotation strategies explained in Chapter 5. A symbolic task could then utilize the vectors to classify unknown nouns into one of the top level WordNet classes used in these experiments.

While this may be helpful, it does not necessarily provide a statisfactory solution for application to smaller corpora (i.e. single documents). Characteristic adjectives are not guaranteed to appear with an unknown noun in a particular document, and the chances decrease with document size. So although the characteristic adjective vectors are moderately inclusive and have a fair number of adjectives, there is still a data sparseness issue related to the frequency of adjectives, especially in smaller corpora. The next chapter will look at ways to extend the idea of differentiation to include not only other parts of speech, but dependency relations as well.

9

Distributionally Derived Symbolic Rules Using Unambiguous Examples

The characteristic adjectives developed in the previous chapter resulted in vectors with much lower dimensionality than typical vector-based similaritiy metrics. Although this makes them more effective than high-dimension vectors on smaller corpora, they can still be hindered by sparse data, especially since they rely exclusively on adjectives. In addition, polysemy in the training corpus dilutes the information content (as with any vector-based approach).

In this chapter I will address these deficiencies by extending the technique used to derive characteristic adjectives to discover symbolic rules for deciding semantic categories. These rules will be based on lexico-syntactic patterns, and like characteristic adjectives, are hypothesized to be highly correlative with a semantic class. I will derive the rules distributionally (as before) and use them to determine the semantic classes represented by the WordNet lexicographer files, or *supersenses* (Ciaramita and Johnson, 2003), extending coverage to words not found in the lexical resource (WordNet).

The work in this chapter will be an integration of distributional and symbolic methods applied to the representation as discussed in Chapter 6. The experiments are based on the approach given in Section 6.3, which suggests incorporating symbolic distributional information into a similarity measure to assign semantic attributes to unknown nouns. In the first stage of the experiments distributional events will be collected from a large external corpus to derive symbolic rules. In the second stage the resulting rules will be applied to the text representation framework to semantically

classify nouns in a test corpus. Although the focus of the chapter is the novel approach of the differentiated rules, I will also touch upon relevant representational properties that effect the implementation.

9.1 Supersense Tagging

The distributional processing used to derive the characteristic adjectives in the previous chapter utilized the semantic classes at the first two levels of the WordNet taxonomy. In this chapter, the semantic classes represented by the WordNet lexicographer files will be used instead. The lexicographer files are used by the WordNet developers when determining semantic properties. Although the lexicographer files are not included in the WordNet distribution, the file identifiers are given for each sense of a word. Each file contains a collection of semantically related synsets organized by concept (e.g. *act*, *animal*, *food*, *process*, *state*), so words having the same file identifier are members of the same conceptual grouping. The files of interest in these experiments (nouns) represent 25 semantic classes which correspond roughly to 7 of the 9 top level nominal nodes in the WordNet taxonomy, combined with a finer classification for the remaining two top-level nodes (*entity* and *abstraction*).

Ciaramita and Johnson (2003) refer to these classes as *supersenses* because they represent broad categories of finer senses. They argue that aggregating the synset information in WordNet into supersense classes not only provides a much richer set of semantic properties (since individual synset properties logically apply to the supersense), but also presents a small corpus of annotated supersense data via the example sentences contained in the synset glosses.

Although the advantages suggested by Ciaramita and Johnson (2003) apply equally well to the top-level nodes of the WordNet taxonomy proper, there are several other reasons for adopting the lexicographer file classification for these experiments. The most important is that the classes represent a more balanced grouping of the taxonomy. For instance, although *entity* and *state* are both top-level nodes, *entity* contains many more child nodes and is thus much more general. Using the lexicographer files, *entity* is represented by finer semantic groupings closer in scope to *state*.

Another advantage of using the lexicographer files is that the fine-grained sense information available in WordNet is smoothed when considering the 25 classes that make up the lexicographer files. Even though a word may have many different senses, each sense is mapped to one of the 25 files and may end up being in the same file as several of the other senses. This effectively reduces the number of senses for a word, depending on the distribution of senses across the lexicographer files, and in some cases makes a polysemous word monosemous with respect to the lexicographer file classes.

Finally, using the lexicographer files also reduces computational complexity. Every sense in WordNet is annotated with its corresponding lexicographer file index, which is much easier to determine than tracing a node's hierarchy to find the root node.

There have been several recent investigations into supersense tagging. Ciaramita and Johnson (2003) demonstrate a multiclass perceptron classifier (Crammer and Singer, 2001) trained on monosemous WordNet 1.6 nouns found in the context of a 40 million-word corpus along with the WordNet 1.6 definitions and glosses. The novel test set they propose is comprised of new words appearing in WordNet 1.7. They report a significant improvement over the baseline heuristic of choosing the most frequent sense (*person*), however the highest accuracy achieved is 52.9%.

Curran (2005) is able to improve this accuracy to 63% using a weighted voting scheme of automatically extracted synonyms. Using a composite 2 billion-word corpus, a vector of shallow grammatical relations was extracted for unknown nouns. This vector was measured for similarity against vectors of known nouns, and a weighted sense score derived from a subset of the most similar nouns was used to determine the supersense of the unknown noun. A hand-coded backoff algorithm was employed for unknown nouns which did not appear in the derivational corpus.

In both experiments, the goal was to assign the correct supersense to a noun which did not occur in the training data. The unknown nouns used in the test set comprised the unambiguous additions to WordNet 1.6 appearing in WordNet 1.7, amounting to 744 unknown noun types. A second test set was derived by withholding 755 noun types from the WordNet 1.6 training data.

The experiments presented in this chapter are also distributionally based and use similar features to those in typical vector-based similarity measures, such as Curran (2005). However, the aim in my experiments is the derivation of symbolic rules for supersense disambiguation of all nouns appearing in a corpus. This encompasses a much larger range of test noun types, since every noun in the test corpus can potentially match a rule context.

The primary application of unknown supersense tagging suggested by Curran (2005) is the automatic extension of lexical resources, such as WordNet. Automatically deciding the supersense categories can aid the automatic or semi-automatic determination of a word's position in the taxonomy.

Knowing a word's supersense can be beneficial for other applications as well. In Chapter 6 I explained how a reliable semantic classification can aid tasks such as co-reference resolution, where attributes inherited from the class can be used to rule out incompatible referents. As demonstrated in Chapter 5, reducing the size of the set of candidate antecedents in this manner can significantly reduce the complexity of co-reference resolution.

For small numbers of high-level classes, such as supersenses, the attribute values associated with each class can be determined manually. For example, the *human/animate/inanimate* attributes correspond to the WordNet *person/animal/object* classes. A finer level of attribute granularity is possible depending on the class granularity. This chapter will focus on the semantic classification task using 25 semantic (WordNet) classes, which can be used as a basis for assigning nominal attributes.

9.2 Symbolic Rules

The characteristic adjective vectors derived in the previous chapter consisted solely of attributive adjectives, i.e. instances of collocated adjective-noun pairs. Although this usually represents the majority of adjective constructions, there are still many more possibilities for distributionally processing adjectives. Additionally, when considering semantic classification, it may be helpful to extend distributional information to include

other parts of speech. For example, Hindle (1990) used *verb-noun* distributional information to classify nouns.

Earlier in the thesis I discussed how symbolic information is increasingly leveraged in distributional experiments. One of the difficulties encountered with this approach is the expense of obtaining accurate symbolic information in large corpora. PoS tags and dependency information are costly to produce by hand and current taggers and parsers cannot achieve complete accuracy, although taggers are generally more accurate than parsers. Additionally, deep syntactic parsers can have high computational complexity, making it infeasible to process larger corpora.

Curran and Moen (2002) evaluate the performance of a distributional task (thesaurus extraction) using context information from shallow syntactic dependency extractors versus a full syntactic parser. They conclude that using shallow processing with reduced computational complexity can be advantageous if it enables a much larger corpus sample than would otherwise be feasible with deeper syntactic processing. This result suggests a trade-off between deep syntactic processing which can recover more relational dependencies but has a higher computational cost, and shallow processing which has limited syntactic coverage but can be applied more easily to larger amounts of data. Although either approach can be applied to deriving symbolic rules, the experiments in this chapter are not meant to be exhaustive and therefore adopt shallow processing techniques which adequately serve to demonstrate the hypothesis.

Because tags are more widely available for large corpora, when attempting shallow distributional processing of symbolic information, it is advantageous to use constructions that can be lexically determined. For example, as the previous chapter demonstrates, simple collocations such as attributive adjectives can generally be determined without dependency parsing. More complex constructions can also be recovered using lexical techniques; e.g. nominal compounds and simple verb phrases are constructions which can usually be recognized without a deep syntactic parse. Although these lexically determined constructions comprise shallow symbolic dependency information, they still represent a syntactic relationship. In contrast, strictly lexical distributional processing typically uses an unordered context window of collocates which treats instances equally, regardless of their type or position. The

experiments in this chapter are based on the following shallow syntactic constructions which extend the attributive adjective vectors from the previous chapter:

1. Attributive adjectives – this includes adjectives immediately preceding the nominal string under study, adjectives in simple conjunctive phrases, and adjectives in a penultimate attributive position. Examples: ***distinguished*** *opera singer,* ***free*** *and confidential service,* ***adequate*** *historical data*
2. Pre-nominal compound modifier – nouns immediately preceding (modifying) the nominal string under study. Example: ***volunteer*** *programme*
3. Simple verbal subject – verbs immediately following the nominal string under study, disregarding intervening auxiliaries and adverbs, including simple conjunctive verb phrases. Example: *the airflow will have* ***increased***
4. Simple verbal object - verbs immediately preceding the nominal string under study, disregarding any intervening adverbs or adjectives, including simple conjunctive verb phrases. Example: *to* ***produce*** *a fluffy Risotto*

Each of these patterns can be used with PoS tags and a regular expression to extract distributional events which can be used to derive symbolic rules, making deep parse dependency information unnecessary. This significantly reduces the complexity of the distributional processing and makes it more feasible to run on large corpora. The trade-off is that more complex constructions will either not be discovered in the distributional data, or will produce spurious events.

After the distributional events are extracted and processed (see Section 9.5.1), the result is a set of symbolic rules that are hypothesized to be strong indicators of a supersense. These rules can be encoded in the representation and applied to text in order to assign supersense information. Encoding the rules for application to the representation takes the same form as distributional dependency queries (see Chapter 6), where constraints are explicitly encoded using the standard representational elements and unconstrained properties are unspecified.

For example, a hypothetical rule derived from the first pattern (1) listed above (i.e. attributive adjectives) would be encoded in CAMEO as

ctx[
obj[**mod**[*adequate*] **class**[]]]

where the specific **mod** element and the unspecified **class** element are used to constrain the form of the **obj** element. This rule representation matches the adjective *adequate* followed by a common noun, as well as various syntactic variations of this construction including a conjunctive adjectival phrase, adverbial modifiers, and compound nouns.

Another hypothetical example, is a rule derived from the last pattern (4) above. The representation of the rule in CAMEO is

ctx[
obj[ID=*n*]
evt[ACTION=*increase* O=*n*]]

This rule will match any verb phrase with a head verb of *increase* and a noun serving as a direct object. It does not constrain any other properties of the verb phrase so that it may appear in constructions such as passive, future progressive, etc. The direct object is similarly unconstrained so it may appear as a proper noun, a group, or any other nominal construction.

The encoding of the rules in the representation is flexible and allows for a range of constraints when applying the rules. The more attributes and elements appearing in the rule representation, the more tightly the constraints imposed on the text matching the rule. The rule derivation processing is responsible for determining which constraints are relevant and encoding the rules with these constraints. Section 9.4 gives the details of the distributional processing used to derive the symbolic rules for these experiments.

9.3 Addressing Polysemy

As I noted previously, polysemy plagues all levels of NLP. This is especially true of distributional processing, where polysemy has the effect of diffusing a word's distributional pattern over possibly unrelated and incompatible contexts. A sense-tagged

corpus would solve this problem, but as with deep symbolic information, a sense-tagged corpus is rare and expensive to produce.

To address polysemy in the distributional experiments presented in this chapter, I will restrict the distributional contexts to non-polysemous nouns. Polysemy can be determined for nouns listed in the WordNet taxonomy and it is trivial to filter these out for a class or group to create a list of non-polysemous words. Using such a list would effectively guarantee that the distributional information extracted from the corpus would not be diluted by polysemy, without requiring word sense tagging.

This approach has been employed in previous research as an alternative to hand-tagged data. Leacock et al. (1998) compare performance of a word sense disambiguator when trained on data automatically derived from monosemous relatives of a polysemous noun versus manually tagged data. Their results are mixed, but much of the performance degradation is due to the assumption that the contexts are interchangeable (which is not the case for polysemous nouns such as *line* and some monosemous relatives such as *picket line*).

Ciaramita and Johnson (2003) use monosemous nouns in the training data of a supersense classifier. After extracting all occurrences of WordNet 1.6 nouns in a 40 million word corpus, they removed all nouns having more than one supersense. They mention that this approach produced better accuracy than including all nouns and assigning distributional information over all senses of a multi-sense noun.

Another common technique involving monosemous words is bootstrapping, i.e. iteratively marking sense information beginning with unambiguous monosemous words. For example, Mihalcea and Moldovan (2000) attempt to annotate an IR system with semantic information. They use a multi-stage processor to disambiguate lexical strings, tagging monosemous tokens at an early stage for incorporation into processing of compositional strings.

An important question to answer before restricting a training corpus to monosemous nouns is whether it will produce enough data. It is reasonable to assume that non-polysemous words are less common than their polysemous counterparts, and therefore will produce fewer instances for distributional processing.

Table 9.3– WordNet 2.1 Noun Database Statistics

unique strings	117,097
Synsets	81,426
word-sense pairs	145,104
monosemous words and senses	101,321
polysemous words	15,776
polysemous senses	43,783
average polysemy including monosemous words	1.23
average polysemy excluding monosemous words	2.77

The WordNet 2.1 statistics for noun polysemy are shown in Table 9.3. Although there is clearly a much larger number of monosemous words, these tend to be more obscure and fall on the tail of a Zipfian-like distribution. Polysemous words are more likely to occur not only because they participate in multiple senses, but also because they are often less specialized. An informal survey reveals that a large majority of the monosemous nouns turn out to be compound or hyphenated nouns. Those that are not are often precise and unambiguous, such as *genuflexion*. Contrast this with typical polysemous nouns such as *man* (11 senses), *hand* (14 senses), and *bank* (10 senses).

Ciaramita and Johnson (2003) report nouns having multiple supersenses accounted for 72% of the tokens and 28.9% of the types in their corpus. The corpus used for the experiments in this chapter has a measured distribution of 35.2% polysemous noun tokens and 13.0% monosemous noun tokens, nearly a 3 to 1 ratio. The remaining objects (i.e. noun phrases) do not have an explicit class (e.g. pronouns and proper nouns).

What effect the distribution of non-polysemous nouns will have on the experiments will depend largely on the size of the classes. If the experiment uses large enough classes the effect of the smaller instances of non-polysemous nouns can likely be mitigated. The following section describes the classes used for these experiments.

Even with this strategy employed to address polysemy, there are other related issues which can affect the results. Metonymy and polymorphism complicate semantic classification. For example, the metonymic usage of groups (corporations, organizations, etc.) as individuals is common in the press. Sentences such as [*The Navy said the accident happened yesterday*] are a simple illustration. The object [*Navy*] would appear in WordNet as a *group* semantically, however the verb *said* is technically related to the semantic class [*person*]. This case is common enough that it warrants conflating the two classes in certain instances. Other cases of metonymy and polymorphism are more difficult to detect. Section 9.5.2 explains how this problem was accommodated in the experimental evaluation.

9.4 Deriving Characteristic Rules

The same principle used to derive characteristic adjectives in Chapter 8 can be employed to discover characteristic rules. Using distributional processing, a set of rules can be extracted for each class, and by determining the differentiae for the rules, a set of unique characteristic rules can be derived. Recall that for characteristic adjectives, the differentiae were calculated relative to siblings sharing a parent node in the taxonomy. This localized approach was necessary because it was not feasible to differentiate globally against all other words, and in addition, the effects of polysemy could result in spurious global matches. For this application, the lexicographer file classes are used instead, which means all the classes are not necessarily at the same level in the taxonomy. However, since there are a smaller number of classes (25 lexicographer files), and the distributional data is non-polysemous, it is feasible to differentiate the derived rules globally (i.e. against all other classes). If a rule matches more than one class, it cannot be considered a characteristic indicator of a single class.

The procedure used for deriving the characteristic rules is summarized as follows. For each class (i.e. lexicographer file), all polysemous words were discarded. The remaining words were used to seed the patterns described in Section 9.2 and collect matching events from the corpus. For example, the lexicographer file representing the class *possession* contains 1,078 entries after removing all polysemous words. Consider one of these, say, *subsidy*. Distributional events for the noun *subsidy* appearing in the

context of one of the patterns described in Section 9.2 would be collected from the corpus. Here are some examples:

1. Attributive adjectives – ***agricultural*** *subsidy*
2. Pre-nominal compound modifier – ***cash*** *subsidy*
3. Simple verbal object - *... she relented and* ***negotiated*** *subsidies to her ex-husband ...*

Matching events were collected for each entry in the filtered class list in this manner. These specific matches were then generalized to form a set of rules, by replacing the seed words with the class and aggregating the rules (removing duplicates).

Once rules for all classes had been distributionally extracted, the differentiae were computed for each rule set by removing all rules occurring in more than one class, resulting in a set of unique symbolic rules for testing membership in each class.

Figure 9.4 illustrates the process using three of the supersense categories. In step 1, distributional events are collected using the list of monosemous nouns for each supersense. Each italicised word in the figure is a monosemous noun which belongs to

Figure 9.4 – Example rule derivations for three supersense classes

	artefact	**body**	**time**
1	the *photo* **appears** …	the *skull* **appears** broad …	**celebrates** the *tercentenary* …
	his *seal ring* **appears** on …	bruised *knuckle* can be **bandaged**	**celebrates** major *anniversaries* …
	the general's *cuirass* **appears** as …	**bacterial** *chromosomes* occur …	the *jubilee* is **celebrated** by …
	the *synagogue* now **exhibits** …	**bacterial** *genes* …	
2	<obj class=*artefact*><verb=*appear*>	<obj class=*body*><verb=*appear*>	<obj class=*time*><verb=*celebrate*>
	<obj class=*artefact*><verb=*exhibit*>	<obj class=*body*> <verb=*bandage*>	
		<mod=*bacterial*><obj class=*body*>	
3	<obj class=*artefact*> <verb=*exhibit*>	<obj class=*body*> <verb=*bandage*>	<obj class=*time*><verb=*celebrate*>
		<mod=*bacterial*> <obj class=*body*>	

the supersense indicated by the heading of the column. The words in bold represent the discovered keywords to use in the rule derivation. In step 2, the lists of distributional events are aggregated and generalized to form supersense rules. Finally, in step 3 the differentiae are computed to produce characteristic rules by removing rules which appear in more than one category. In this example, the verb *appear* is found in rules for both the **artefact** and **body** supersense categories so all rules using *appear* are discarded.

Although the rules are derived using shallow lexico-syntactic pattern matching, the application of the rules in the text representation allows for them to be used with rich syntactic dependency constructions. As an illustrative example, consider the following characteristic rule produced for the *object* class:

<verb=*shield*><obj class=*artifact*>

This rule states the syntactic object of the head verb *shield* is of the semantic class *artifact*. The following sentence in the Wolverhampton corpus matches the rule:

Handles should be shielded with rubber.

The CAMEO representation is given by:

ctx [TYPE=*clause*
obj [ID=*o11* **class** [*handles*]]
evt [O=*o11* ACTION=*shield* MODAL=*should* PASSIVE] **rel** [*with rubber*]]

The rule classifies the object [*handles*] correctly as a member of the *artifact* group. Note that although the verb phrase is expressed in passive form, it is normalized in the symbolic CAMEO representation allowing the rule to match correctly. A strictly lexical application of the rule would either miss this instance, or incorrectly interpret *handles* as the subject.

9.5 Experiments

As with the experiments in the previous chapter, there are two major parts to the experiments described here. First, deriving the set of characteristic rules according to the procedure described in Section 9.4. Second, using this derived set of rules to decide the semantic classification of nouns. The next two sections will discuss each of these parts in more detail.

9.5.1 Distributional Processing

In order to ensure the rules derived through distributional processing are unique (i.e. characteristic), a large, tagged derivation corpus is needed, such as the BNC used in the previous chapter. Due to time and resource constraints, rather than using the entire BNC a sub-corpus comprised of all files in the BNC A section was used for these experiments. The BNC A section is comprised of 674 written documents and contains 14,232,256 (orthographic) words, which represents approximately 15% of the BNC corpus.

The derivational corpus is processed using distributional seeds from the WordNet 2.1 lexicographer files. Each WordNet 2.1 lexicographer file contains a list of "words" which can include multi-word expressions and hyphenates. Hereafter I will refer to these as "strings", following the WordNet nomenclature. After removing all polysemous (multi-class) strings, each lexicographer file yielded a set of monosemous (single-class) strings. Table 9.5.1 lists information about each lexicographer file used in the experiment. The first column gives the WordNet lexicographer file index, the second column gives the file name, and the third column gives the number of (monosemous) strings after removing polysemous strings. The table is sorted by the number of strings from highest to lowest.

The distribution of (monosemous) strings over the semantic classes exhibits a very clear Zipfian form. The first four classes comprise nearly half of the distribution, with the remaining strings distributed in a roughly logarithmic curve. This is due to the nature of the first four semantic classes, which are all tangible and comprise the majority of the nouns in WordNet: *plant, person, artifact, animal*. Although other

tangibles appear further down in the table, they are more specific, such as *body* and *food*, resulting in smaller classes.

Table 9.5.1

Distributional statistics for derivation corpus by class/lexicographer file

File	Name	Strings	Events	Rules	Pre	Adj-1	Adj-2	Obj	Subj
20	plant	16,865	5393	556	66	140	28	31	48
18	person	16,322	132,925	4,489	914	1,717	549	382	58
6	artifact	13,375	80,936	4,834	1,123	1,758	347	350	272
5	animal	13,351	11,419	598	106	226	27	46	21
4	act	6,193	44,658	1,350	264	557	112	102	43
10	communication	5,638	56,243	1,544	342	734	126	113	41
26	state	4,105	17,871	479	58	189	46	35	91
27	substance	3,968	13,734	1,049	193	338	61	72	5
15	location	3,936	67,871	407	69	140	44	27	8
8	body	3,035	6,807	315	60	106	26	35	28
7	attribute	3,007	22,479	444	47	199	39	41	81
14	group	2,961	39,006	886	195	305	94	69	34
9	cognition	2,863	22,163	690	129	369	72	53	0
13	food	2,468	9,929	594	147	194	51	57	27
17	object	1,682	14,986	1,132	150	224	29	63	528
23	quantity	1,483	7,608	123	26	18	4	6	9
28	time	1,355	29,594	313	42	64	12	16	26
21	possession	1,078	15,605	447	87	145	28	34	22
11	event	785	6,504	203	54	64	22	11	2
22	process	749	2,467	51	10	19	4	3	3
19	phenomenon	684	7,553	88	14	35	7	6	0
24	relation	448	969	26	6	12	0	3	2
12	feeling	340	4,556	127	12	75	9	10	18
25	shape	255	825	81	4	12	4	3	54
16	motive	50	127	21	0	2	0	0	19
	Total	106,996	622,228	20,847	4,118	7,642	1,741	1,568	1,440

The strings in each file were used to filter sentences from the derivation corpus. Any sentence which matched one of the strings in the file was extracted, yielding a set of distributional events for each filtered lexicographer file. The Events column of Table 9.5.1 records the number of events produced by each file.

Note the distribution of events does not follow the distribution of strings in the lexicographer files. The largest number of events occurs for the second largest class

(*person*). The largest ratio of events to strings occurs with the much smaller *time* class which yields 29,594 events from only 1,355 strings.

Each set of events was processed for patterns matching those described in section 9.2, where the nominal string under study was taken to be a string from the corresponding filtered lexicographer file. This yielded a set of (possibly overlapping) symbolic rules for testing membership in each class (lexicographer file grouping). Within each class, the rules were generalized by abstracting the nominal string used in the match and removing duplicates (see Section 9.4).

The final step in the distributional processing was to calculate the differentials in the symbolic rule sets by removing all duplicate rules globally across all classes, leaving a set of unique rules hypothesized to be characteristic for a given class. The Rules column in Table 9.5.1 gives the total number of characteristic rules produced for each class. The subsequent columns list the number of rules by category, where:

Pre – pre-nominal modifier

Adj-1 – attributive adjective (first position)

Adj-2 – attributive adjective (second position)

Obj – verb phrase object

Subj – verb phrase subject

Most of the classes retained less than 10% of their events as rules after differentiation, except *motive* (16.5%) and *plant* (10.3%). The smallest percentage of rules retained was *location* (0.6%), and the average percentage of rules retained was 3%.

The last line in Table 9.5.1 gives the totals across all files. After differentiating the rules there remained 20,847 rules, with the largest share of these derived from attributive adjectives (7,642). Compound nouns also accounted for a large number of these, and verb-based rules were the least common. (See Section 9.6 for a discussion of the distribution of verbs).

9.5.2 Evaluation

Unlike the characteristic adjectives derived in the previous chapter, the symbolic rules being tested here are not vector based and subsequently have a more narrow scope, resulting in fewer instances where each rule will apply. On the other hand, the semantic classes being decided are broader and more comprehensive (since they represent a flattened portion of the WordNet hierarchy), which means that there are many more class members which form the distribution.

I tested the derived rules on several corpora: The MUC7 corpus, the Wolverhampton corpus, and the Siddharthan corpus. Each corpus was processed using the RASP toolkit and transformed into the CAMEO text representation language. The test set of nouns was selected as follows. All nouns in the corpus found in one of the syntactic constructions described in Section 9.2 were collected. From this set, all first and second pronouns were removed, along with all monosemous nouns. The resulting set of test nouns represented the majority of all ambiguous nouns (with respect to supersenses) in the corpus.

The derived rules were transformed into queries which were applied to the set of test nouns from each corpus, resulting in a set of noun phrases and their classification. For evaluation, the correct classification was manually determined from the set of senses listed in the WordNet database for the head noun. This included a consideration for the context of the noun, and whether its usage involved metonymy or polymorphism.

For example, the Siddharthan corpus includes a series of fables which include characters such as talking animals. In this case, the animal will exhibit both human and animal behaviour and could be classified as *person* or *animal* depending on the context. That is, when the character is exhibiting human behaviour it should be classified as *person*, and in all other cases it should be classified as *animal*. A further example is from a newspaper article about the terrorist group Al-Qaida. Normally this noun phrase would be assigned the *group* class, however the sentence [*Al-Qaida will feed*] invokes a metaphoric reference and warrants an *animal* class in this context. A similar argument can be made when determining the classification of metonymic usages of *group* for *person*.

These types of usage accounted for a relatively small percentage of the overall test cases matched by the rules. The Siddharthan corpus contained the greatest number (4.9%), while fewer instances were found in the MUC7 (3.7%) and WLV (0.4%) corpora. The percentage across all corpora combined amounted to just 2.6% of the test cases matched by the rules.

The baseline algorithm used for comparison consisted of selecting the first listed WordNet (super) sense of an object's head noun. The WordNet documentation states that the first listed sense is the most frequent with respect to the WordNet development corpus. Using the most frequent sense of a word is a typical approach for dealing with polysemy in distributional experiments. For objects that do not include a common noun, such as proper nouns and pronouns, this approach does not apply, and these types of objects were not included in the calculation of the baseline precision.

The following definitions are used to describe the baseline scoring metrics:

N is the set of noun phrases under test (as described above)

N_B is the subset of noun phrases in N having a common noun (i.e. classifiable by the baseline algorithm).

N_C is the number of correctly classified noun phrases

The precision *P* and recall *R* are given by:

$$P = {N_C}/{N_B}$$

$$R = {N_B}/{N}$$

The F-measure is calculated in the standard way:

$$F = \frac{2PR}{P + R}$$

Extending the baseline approach of first-listed sense, the characteristic rules provide an alternate means to decide the supersense of nouns, increasing the coverage to include nouns not contained in N_B. The algorithm applies the rules to all noun phrases in the set N defined above. For noun phrases matching a characteristic rule, the rule is used to select the target classification. For cases where the set of senses listed in Wordnet for the noun phrase does not include the rule's target classification, or there are multiple rule matches ascribing conflicting supersense classes, the algorithm reverts to the baseline (i.e. most frequent sense). The characteristic rules are also applied to noun phrases that do not contain common nouns (and cannot be classified by the baseline).

The following definitions are used to define the scoring metrics for the characteristic rules:

N_R is the subset of noun phrases in N classifiable by a characteristic rule.

N_C is the number of correctly classified noun phrases

F is defined as before. The precision P and recall R are given by:

$$P = {}^{N_C}/_{N_R}$$

$$R = {}^{N_R}/_{N}$$

9.6 Results

An important goal of these experiments is the evaluation of the representation itself and the properties which facilitate deriving and applying the symbolic rules. Although a quantative measure would be difficult to devise, a few conclusive results can be made through analysis. First, the normalisation of the syntactic structure afforded by the representation is conducive for applying symbolic rules of this type, allowing a single rule form to apply to a wide range of syntactic constructions such as passive verb phrases, subject-auxilliary inversion, complex verb phrases, verbal and nominal conjunctions, and infinitival complements.

Next, the structure of the representation and its contextual organization aid in certain corpus processing tasks conducted during the course of the experiments, such as the alignment of several corpora. For example, during the development of the experimental framework, several iterations of the representation were produced from the corpus due to issues in the early stages of processing that needed to be resolved. Transferring the annotation key to the successive iterations of the representation was achieved through a simple transform which aligned sentential and phrasal contexts before attempting to align individual objects. Information from the key corpus was then copied into the object containers for those objects that aligned with the new representation. (Any remaining objects were subsequently re-annotated.)

Finally, the object-centric nature of the representation is advantageous for generating reports and analysis. The representation of noun phrases as first-class objects in a phrasal context makes them more accessible during processing than strictly hierarchical representations. Also, because the representation of objects is uniform, regardless of the underlying grammatical relations, transformational processing can be applied globally to objects as a class. Generating global lists of objects with specific properties or relations can be helpful for analysis, as for example the evaluation of ruleset classes in the experiments of this section based on an object's syntactic position (Figure 9.6-2).

Table 9.6-1
Precision, Recall and F-measure for semantic classification on three corpora

	Characteristic Rules			**Baseline**		
	P	**R**	**F**	**P**	**R**	**F**
MUC7	60.4% (808/1337)	88.8% (1337/1505)	71.9%	61.7% (641/1039)	69.0% (1039/1505)	65.2%
Siddharthan	60.7% (1740/2867)	86.9% (2867/3298)	71.5%	62.5% (1549/2480)	75.2% (2480/3298)	68.2%
WLV	50.8% (1911/3759)	87.7% (3759/4284)	64.4%	50.8% (1860/3664)	85.5% (3664/4284)	63.7%
Total	56.0% (4459/7963)	87.6% (7963/9087)	68.3%	56.38% (4050/7183)	79.1% (7183/9087)	65.8%

Table 9.6-1 shows the qualitative results of applying the baseline and derived symbolic rules to three sets of corpora. Using characteristic rules improved the F-score over the baseline for each corpus, with the largest improvement recorded for the MUC7

corpus. The characteristic rules improve the recall by increasing the coverage of the baseline system to include nouns which do not occur in WordNet (the majority of these being proper nouns and certain pronouns). The recall improved by 10% for two of the corpora (MUC7 and Siddharthan), with an average improvement over all three corpora of 8%. The WLV corpus had much higher recall for the baseline than the other corpora (86%) and this resulted in a smaller improvement when the characteristic rules were applied. The WLV corpus is comprised of mainly instructional documents which contain a smaller ratio of non-common nouns. This contrasts with the MUC7 corpus which is newspaper text and has a much higher ratio of proper nouns and pronouns resulting in a lower baseline recall (69%).

The precision of the characteristic rules was similar to or slightly worse than the baseline. One factor that affected the performance was processing errors. Parsing errors (which were often the result of tagging errors) accounted for roughly 3-5% of the incorrect classifications (based on a sample analysis). For example, sentence (100) shown below is from the WLV corpus.

(100) *Do not use hammers to bash wooden or plastic handled tools such as chisels or screwdrivers.*

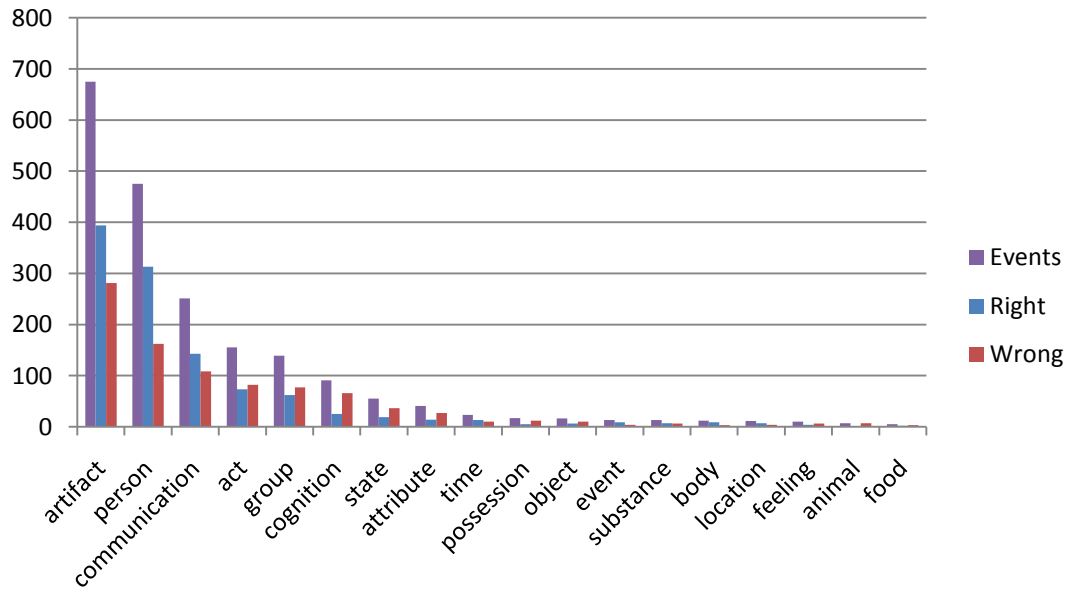
In this case, the syntactic processing mistagged/misparsed *plastic handled tools* resulting in the misapplication of the characteristic rule <verb=*handle*><obj class=*act*> and the erroneous supersense *act* assigned to *tools*.

Another factor affecting the precision is the variation in the supersense categories. Certain objects are given a classification in WordNet which does not always reflect the more common usage. For example, the WLV corpus contains several documents which discuss software programs and operating systems. In WordNet, these types of objects are classified as *communications*, whereas a large number of the characteristic rule applications resulted in the supersense *artefact*. In many cases, the collocates associated with the *communications* supersense, such as *broadcast*, *receive*, etc., do not apply to software programs, but those of the *artefact* supersense often do (e.g. *install*, *use*). This ambiguity accounted for a large percentage of the errors in the WLV corpus. For example, 38% of the errors in the Linux HOWTO document, and 67% of the errors in the CDROM HOWTO document were due to this ambiguity.

Figure 9.6-1 shows the distribution of matching events and the rule accuracy for each class in the test corpora (classes with less than 1% of the matches have been omitted). As expected, based on the distribution of derived rules, the largest class was *person* followed by *artefact*. These two classes combined accounted for over 50% of the events in the three test corpora. Although *person* had the second largest number of monosemous strings in WordNet (behind *plant*), it still made up the largest percentage (21.3%) of the distribution of events in the derivation corpus. This frequency of the *person* class in the derivation corpus appears to correlate with the test corpus, accounting for the high number of events matching the *person* class rules. The *artefact* class, which had 20% fewer monosemous strings than *person*, accounted for 13% of the events in the derivation corpus, and exhibits a proportional distribution in the test corpus. Note that *artefact* produced a larger number of rules than *person*, and still had fewer events in the test corpus.

The rule accuracy was better for *person* (65.9%) than *artefact* (58.4%), with the less frequent class accuracies varying widely. Only *body* had better accuracy (75%) than *person/artefact* and *body* appeared in significantly fewer events.

Figure 9.6-1
Event distribution and rule accuracy by class



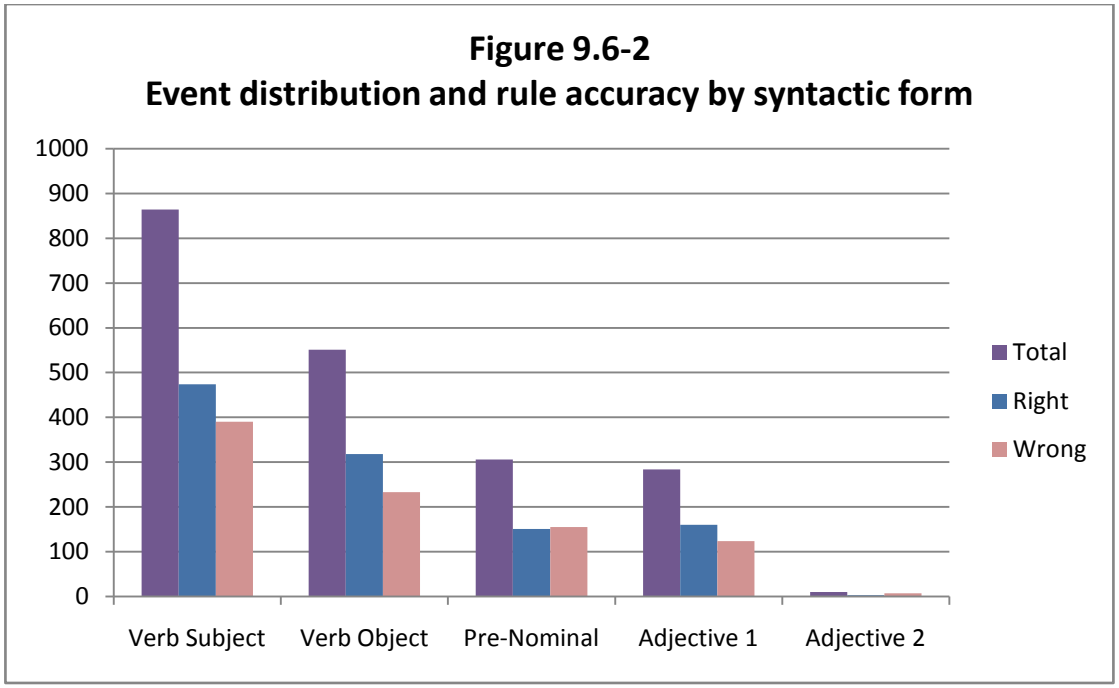


Figure 9.6-2 shows the event distribution over the test corpus of matching rules by syntactic form. The *verbal subject* rule form clearly dominates the distribution. This is a surprising result because the verbal subject form accounts for just 7.5% of the total number of derived rules. In fact, the largest percentage of derived rules (36.7%) takes the *adjective-1* form, and this produces a relatively small number of matches in the test corpus.

One possible explanation for the larger number of matching verbal rule events is the underlying distribution of verbs in the derivation corpus. According to statistics compiled by Leech et al. (2001), the ratio of noun to verb *tokens* per million words in the BNC is roughly 1:1 (181,985 / 174,272), while the ratio of noun to verb *types* is roughly 3:1 (3,031 / 1,103). This suggests that far fewer unique verbs than nouns will appear in a given corpus. Thus any derived verbal rules would most likely represent a larger number of matched verbal events in the derivation corpus, and subsequently generate a proportional number of matches in the test corpus.

Conversely, the distribution of matching adjective events (11.9%) appears relatively equal with nouns (13.9%) in the test corpus distribution. Even though adjectives have a WordNet ratio to nouns of roughly 5:1, they have a much lower distribution in running text. Leech et al. (2001) report BNC adjective statistics per million words at 55,328 *tokens* and 1,036 *types*, equating to a roughly 3:1 noun-to-adjective *type* ratio (the same

as noun-to-verbs) and a 3:1 noun-to-adjective *token* ratio (much higher than noun-to-verbs). However, only a small percentage of nouns will appear in the compound form and this produces a similar percentage of rules (adjectives 45%, nouns 40.1%), resulting in a similar distribution of matching events observed in the test corpus.

Unlike statistically based approaches, using unambiguous monosemous words to seed the rule derivation avoids the affects of polysemy and allows the algorithm to work with a relatively small number of distributional samples. The average number of events observed in the derivation corpus for a particular monosemous word in a given rule pattern was 1.2, which is extremely small in comparison with the numbers of events used in the experiments on adjectives from the last chapter. Even when the events are smoothed to create a rule, the aggregate numbers remain relatively small. This is a direct result of using "pure" monosemous instances, which do not appear as frequently as polysemous words. Of course this does not guarantee that the derived rule will not appear in a polysemous context within the corpus. But the experiments show that some rules can be derived which possess a close semantic correlation with the class of the monosemous words used to derive them.

Table 9.6-2
Examples of derived rules

Class	Pre-nominal	Adjective-1	Verb Subject	Verb Object
act	*bombing*	*pre-emptive*		*accomplish*
artefact	*cockpit*	*high-speed*	*sink*	*board*
cognition	*retrieval*	*hegelian*	*preclude*	*solve*
communication	*cancellation*	*conversational*	*pre-date*	*answer*
food	*caramel*	*piquant*	*thicken*	*simmer*
location	*plateau*	*hilly*	*landlock*	*stretch*
person	*predecessor*	*itinerate*	*understand*	*oblige*

Table 9.6-2 lists examples of the derived rules for several classes. In most cases the rules are highly suggestive of the appropriate class. However, it is clear that counter examples can be constructed where the rule could fail (as is true for all distributional processing). But because the rules are only applied to matching contexts where the rule's supersense is listed in WordNet as one of the noun's supersenses, only certain

nouns appearing in these constructions will match the rule. This raises the rule's precision at the expense of lowering the recall.

9.7 Conclusion

Distributional processing is often used to derive statistical measures, as for example in Chapter 7. However, in this chapter I have demonstrated a distributional approach to deriving symbolic rules. The rules represent an aggregate of distributional instances in a derivation corpus but do not include an explicit statistical weighting. Instead, the same process of differentiation, introduced in the last chapter, is applied to filter ambiguous rules, leaving only those which correlate with a single class in the derivation corpus.

In order to derive single-class rules, the effects of polysemy must be addressed. In this chapter I used known monosemous words to avoid the need for sense-tagged data. This ensures that the events observed in the derivation corpus have the desired class. However, this does limit the data used in the rule derivation to events occurring with the monosemous words.

The results demonstrate that this technique is able to extend the baseline approach (of defaulting to the first-listed WordNet sense) to words that do not occur in WordNet. The experiments derived many rules for disambiguating noun phrases using the process of differentiation to select rules which are indicators of a single semantic class. Because the rules are syntactic in nature, each applies to a specific syntactic context. The coverage of the ruleset is directly related to the number of rules derived, and their lexical contexts. The experiments demonstrated a nearly 10% gain in recall when the baseline was extended with the characteristic rules applied to unknown nouns.

The precision of the rules remained at, or slightly lower than, the level of the baseline. Part of this was due to errors in processing, but semantic factors also came into play. Metonymy, analogy, and other creative uses of language need to be accounted for when applying symbolic rules of this type. The characteristic rules derived in the experiments often were highly predictive, but sometimes the rule matched a lexical context quite different than those used to derive the rule. In many of these cases the

correct supersense was a closely related class and smoothing these classes might be an alternative approach, depending on the task. Other possibilities for refining the rules include considering a wider context (e.g. both verbal subject and object), and using distributional information about polysemous contexts where the rule appears (in the derivation corpus).

As with any distributional approach, the derivation corpus determines the quality of the distributional output. The experiments reported in this chapter made use of a sub-corpus taken from the BNC which is large enough to provide an adequate sample of text. Using a larger corpus could possibly result in more events, which in turn could generate more rules, but these would likely be increasingly rarer usages having little effect on the overall results.

10

Conclusion

In this thesis I have investigated the integration of contextual and distributional processing with a sematically-motivated linguistic representation. The objectives of the thesis were set out in the introduction and the thesis has addressed each of these in turn: 1) I have defined and implemented a text representation language called CAMEO, which satisfies the proposed desiderata and provides the framework for a general treatment of contextual and distributional processing at all levels of discourse; 2) I have developed a systematic treatment of structural and linguistic context within the representation and applied it to a text processing task; 3) I have investigated distributional and statistical methods of text processing and developed a novel distributional application to symbolic processing.

The main contributions of the thesis are a novel representation of context, a distributional approach to deriving symbolic rules, and several innovations which distinguish the text representation language. In addition, using semantic properties of adjectives, I have developed a novel distributional semantic similarity measure, and proposed a new approach to deriving a gold standard for evaluating statistical lexical acquisition strategies.

The CAMEO representation language developed in Chapter 3 provided the framework for exploring issues of context and distributional processing in the thesis. To guide the design of the representation language I have explored several aspects of representing linguistic analysis which affects a wide range of language and discourse processing tasks. The different linguistic information required by these various tasks and the wide range of existing representational forms, spanning linguistic and semantic analysis, suggests the need for a flexible representational strategy which is capable of encoding multiple levels of linguistic analysis while supporting deterministic

transformations into other linguistic and semantic forms. To this end, I have proposed desiderata in Section 1.1.1 for a generalized intermediate representation for text processing, and used these desiderata to define the CAMEO text representation language.

CAMEO is a comprehensive approach to an intermediate representation, developed for the research objectives of this thesis. However, it embodies representational principles which can be adapted to other representations, or help when designing new languages. Text representations in the literature often take the form of ad-hoc task-specific approaches, or general languages designed to support a particular aspect of language processing. In both cases it is rare to find representational issues addressed directly and a comprehensive treatement of lexical and syntactic variations in the representation. Recently, as discussed in Section 1.1.1, representational issues have gained more interest due in part to the work of the parser community's attempt at deriving a common ground for evaluating different parsing systems. Other representational concerns have surfaced such as the complexity posed by some representations to non-linguistic users. Many of these concerns are encompassed in the desiderata proposed in this thesis and implemented in the CAMEO language. Chapter 3 explores the properties and form of the representation, and gives a comprehensive account of syntactic variations. In Chapter 4, examples of operations on the representation demonstrate its usability and support the claim that it provides a form that is amenable to manual manipulation and analysis.

CAMEO is a linguistic encoding with rudimentary semantic types, and includes several features and properties not found in extant representations. Besides the integration of contextual and distributional information (discussed below), CAMEO includes a strategy for the representation of heterogenous groups (Sections 3.3.1.1and 3.4.9), possessives/genitives (Section 3.4.4), and reflexives (Section 3.4.8). It also includes a general representation for arbitrary bracketing in conjunctive constructions for constituents such as adjectives, nouns, and phrases (Section 3.4.2), and supports a primitive meaning representation suitable for shallow semantic tasks (Section 3.1.7).

CAMEO is positioned at the intermediate linguistic level and captures a wide range of linguistic information from the surface text and analysis. Chapter 4 explores surface realisation from the internal representation as a means of testing the richness of the

encoded information. Rather than relying on a grammar, surface realisation in CAMEO is achieved through a direct, deterministic transformation. This gives shallow tasks that produce surface variations the ability to manipulate the representation directly without extensive linguistic knowledge, as demonstrated in Section 4.2. The surface realisation transformation is recursively able to realise all levels of the representation, from individual constituents to document level structure.

Operating at the intermediate level allows CAMEO to encode some linguistic functions for reuse across a range of tasks. In Section 4.2, applications to sentence condensation and activisation demonstrated how syntactic variations are normalised and encoded in the representation. The passive/active verb alternation described in the experiments is one example of a linguistic function encoded in CAMEO which presents a canonical representation and alleviates tasks from supporting both forms.

Most textual representations of linguistic analysis function at the sentence level, and do not encompass analysis of larger linguistic units. To extend the CAMEO language for use beyond sentence and phrasal analysis, in Section 3.5 I examined representational issues with respect to discourse- and document-level processing, and quantified these in terms of requirements on a language processing system. I defined data structures in CAMEO to address these requirements, including elements for class-based and assertional processing. Central to the design of these structures is a general and recursive representation of contexts to encode document structure that extends to the sentence and phrasal levels.

In Chapter 5 I looked at some contextual issues affecting one application of symbolic NLP (coreference resolution) where the representation of context can be of benefit. Using the strictly linguistic contextual information extracted from the discourse and syntactic structure, I showed that the representation provides a means for addressing certain aspects of the task which suffer under non-contextual processing. The influence of context on analysis varies with the genre and composition of a corpus, and I presented several experiments to measure and test contextual processing on different styles of corpora.

The representation of distributional information often takes the form of vectors of event counters, requiring little beyond a task-specific implementation. However, as

distributional methods increasingly make use of symbolic information and are incorporated into symbolic processing, representational issues warrant consideration. In Chapter 6 I described how the CAMEO representation language supports augmenting language tasks with both symbolic and distributional information. I explained how distributional information beyond typical collocations, including symbolic dependencies, can be derived from the intrinsic representation using structured queries having explicit constraints, and explored applications of this type of data on symbolic tasks. I also described the support in the representation for distributional data derived from external sources, such as large derivational corpora with shallow representations, and gave examples of how this could be applied.

One of the more common distributional applications of large derivational corpora is lexical acquisition. In Chapter 7 I explored the general properties of statistical similarity measures, which are a critical component in lexical acquisition. I presented a study of statistical methods and several experiments using adjectives, which comprise a lexical category less commonly employed with lexical acquisition and thus of further theoretical interest. As part of the study, I investigated some of the issues with evaluating existing measures of semantic similarity used in acquisition, and proposed a novel objective solution using adjective antonyms as a gold standard. Using antonyms avoids the expense of manually annotated reference corpora, since lists of antonyms are available, without resorting to artificial terms. The experiments in Chapter 7 employed a sample of accepted antonyms to evaluate three extant statistical similarity measures, providing qualitative information about their performance.

In Chapter 8, I suggested several difficulties with using external distributional information from lexical acquisition to augment language tasks. To address these issues, I investigated the use of adjective distributions in the semantic classification of nouns. I have proposed linking adjectives to conceptual attributes and developed the idea of *characteristic adjectives* as differentiating a nominal node in a semantic taxonomy. I have shown experimentally that distributionally derived vectors of characteristic adjectives correlate more closely with a semantic node than a corresponding vector of undiscriminated adjectives. These vectors are much smaller and semantically focused, making them more suitable for augmenting language processing tasks.

Finally, in Chapter 9 I showed how differential distributional techniques used to develop the statistical vectors of characteristic adjectives could be extended to derive symbolic rules. Using differential distributional processing of shallow syntactic information I extracted a set of highly-correlative rules for classifying nouns into coarse semantic classes, called *supersenses*. The assigned classes can be used to map attributes onto the nouns for use in processing tasks such as coreference resolution. In the experiments, the derived symbolic rules were shown to improve the recall of a baseline system by extending coverage to nouns not encompassed by a lexical resource. A notable feature of this approach was the use of monosemous words to avoid the problems of polysemy. Only words having a single class for all senses (which can be determined from the lexical resource) were used in the distributional processing, avoiding the need for a sense-tagged corpus.

10.1 Further Work

There are several areas covered in the thesis which suggest further lines of research. First of all, the text representation language currently does not include an account of underspecification like that found in some semantic representations. As I explained earlier, the treatment of universal quantifiers by the representation makes this unnecessary, since they retain their surface representation. However, a more relevant issue of ambiguity, with respect to a linguistic analysis, is attachment ambiguity of constituents such as prepositional phrases. Extending the text representation to allow for underspecifying ambiguous syntactic attachments (cf. with packed parse tree structures) would simplify the job of the parser, while deferring the resolution to deeper processing that may have accesses to more linguistic resources.

Another area of interest is extending the language processing system to other parsing technologies. The current system has been implemented on two parsing systems, focusing on the RASP system. By implementing transformation modules for outputs of other different parse technologies, certain deficiencies in the representation would likely become manifest. Addressing these might improve the application of the system as a general intermediate representation. A similar argument can be made for implementing new tasks using the language processing system. Tasks which provide useful annotation

for deeper processing, such as named entity recognition, would provide a further useful benefit.

In addition to extending the application of the representation, further work could be done to enhance its usability. Although the internal form of the representation makes it practical to manipulate by hand, providing automated tools could simplify the process. For example, it would not be difficult to develop a "drag and drop" interface to create and edit representations, using off-the-shelf components.

One possible enhancement to the existing methodology which might yield improvements in the internal representation is further testing of surface realisation. The surface realisation task in Chapter 4 was used to test the expressiveness of the representation by encoding sentences in the representation and processing them through the surface realiser. Although this included a range of syntactic constructions, it was not a systematic test. Further work on developing a more systematic approach may prove beneficial.

The evaluation of statistical similarity measures using adjective antonyms also warrants further research. Experiments involving larger lists of antonyms, possibly including unrelated adjectives might yield more insights into the operation of the various similarity measures. Other types of similarity measures could also be tested in this manner, for example machine learning approaches.

An interesting possibility for refining the set of characteristic adjectives would be to use a weighting based on the frequency of adjectives. Currently only the percentage of matches is used when selecting for membership. It may be possible to improve the performance by allowing weighted matches based on the relative frequencies of observed occurrences of adjectives in the characteristic set.

Though it may not provide any new insight to the representation language, further improvement of the coreference resolution algorithm would be useful. Coreference is a fundamental requirement of deeper processing such as semantic tasks, therefore extending the research on coreference resolution could be leveraged for other research. In particular, other resolution algorithms (e.g. Lappin and Leass, 1994) could be applied

independently or in parallel. Also, further experimentation of supplemental symbolic and distributional methods could be explored.

Perhaps the most important area to be explored is the potential for integrating multiple analyses at various levels of processing, which is afforded by the representation. Probabilistic components can usually be configured to return multiple ranked analyses and incorporating analyses from competing components are both possibilities using a unified intermediate representation and warrant investigation.

The wide range of research suggested by the representation and its applications to contextual and distributional processing helps to underscore the integral role aspects of a representational language can play in natural language processing.

Bibliography

Abney, Steven. (1996). **Statistical Methods and Linguistics**. In *The Balancing Act*, Judith Klavens and Philip Resnik, editors, MIT Press, Cambridge, MA.

Allen, James. (1984). **Towards a general theory of action and time.** *Artificial Intelligence*, 23(2), pp. 123-154.

Allen, James. (1995). *Natural Language Understanding*. The Benjamin/Cummings Publishing Company, Inc., Redwood City, CA.

Alshawi, Hiyan, ed. (1992). *The Core Language Engine*. MIT Press, Cambridge, MA.

Alshawi, Hiyan and Richard S. Crouch. (1992). **Monotonic semantic interpretation**, In *Meeting of the Association for Computational Linguistics*, pp. 32-39.

Alshawi, Hiyan. (1996). **Head Automata and Bilingual Tiling: Translation with Minimal Representations**, *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL-94)*, Santa Cruz, CA, pp. 167-176.

Aone, Chinatsu and Scott Bennett. (1996). **Applying machine learning to anaphora resolution**, In Stefan Wermter, Ellen Riloff, and Gabriele Scheler, editors, *Connectionist, statistical and symbolic approaches to learning for Natural Language Processing*, Springer, Berlin, pp. 302-314.

Argamon, Shlomo, Ido Dagan, and Yuval Krymolowski. (1999). **A Memory-based Approach to Learning Shallow Natural Language Patterns**, *Journal of Experimental and Theoretical AI*, Vol. 11, pp. 369-390.

Bergler, Sabine. (1995). **From lexical semantics to text analysis**. In Saint-Dizier (1995), Ch. 5, pp. 98 – 124.

Berstel, Jean and Luc Boasson. (2000). **XML Grammars**. In *Lecture Notes in Computers Science*, Volume 1893, Jan 2000. Springer, Berlin., pp. 182 - 191

BNC (2002). *The British National Corpus*. Oxford University Computing Services, Oxford. http://www.hcu.ox.ac.uk/BNC/

Bos, Johan , Björn Gambäck, Christian Lieske, Yoshiki Mori, Manfred Pinkal, and Karsten Worm. (1996). **Compositional semantics in Verbmobil**, *Proceedings of the 16th International Conference on Computational Linguistics (COLING '96)*, pp.131-136.

Bos, Johan. (1996). **Predicate logic unplugged**, *Proceedings of the 10th Amsterdam Colloquium*, pp. 133-142.

Brill, Eric., David Magerman, Mitch Marcus, and Beatrice Santorini (1990). **Deducing linguistic structure from the statistics of large corpora**. *Proceedings of the DARPA Speech and Natural Language Workshop*, San Mateo, CA. Morgan Kaufmann, pp. 275-282.

Briscoe, Ted and John Carroll. (2002). **Robust Accurate Statistical Annotation of General Text**. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, Las Palmas, Gran Canaria, pp. 1499-1504.

Brown, Peter F., Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. (1991). **Word-sense disambiguation using statistical methods**. *Proceedings of the 29th Meeting of the Association for Computational Linguistics (ACL '91)*, pp. 264-270.

Bunt, Harry. (2003). **Underspecification in semantic representations: Which technique for what purpose?**. In *Proceedings of the 5th Workshop on Computational Semantics (IWCS-5)*, pp. 37-54.

Burton-Roberts, Noel. (1999). *Analysing Sentences*. Longman, London.

Buvač, Sasa. (1996). **Resolving lexical ambiguity using a formal theory of context**. In Kees van Deemter and Stanley Peters, editors, *Semantic Ambiguity and Underspecification*, CSLI Publications, pp. 101-124.

Canning, Yvonne Margaret. (2002). **Syntactic Simplification of Text**. *Phd Thesis*. University of Sunderland.

Carletta, Jean, Jonathan Kilgour, Tim O'Donnell, Stefan Evert, and Holger Voormann. (2003). **The NITE Object Model Library for Handling Structured Linguistic Annotation on Multimodal Data Sets**. *Proceedings of the EACL Workshop on Language Technology and the Semantic Web (3rd Workshop on NLP and XML, NLPXML-2003).*

Carroll, John, Ted Briscoe, and Antonin Sanfillipo. (1998). **Parser Evaluation: A Survey and a New Proposal**. *Proceedings of the 1st International Conference on Language Resources and Evaluation*, pp. 447-454.

Chandrasekar, Raman, Christine Doran, and B. Srinivas. (1996). Motivations **and Methods for Text Simplification**. In *Proceedings of the Sixteenth International Conference on Computational Linguistics (COLING '96)*, pp. 1041-1044

Charles, Walter G. and George A. Miller (1989). **Contexts of antonymous adjectives**. *Applied Psycholinguistics* 10, pp. 357-375.

Chen, Jen Nan and Jason S. Chang (1998). **Topical clustering of MRD senses based on information retrieval techniques**. *Computational Linguistics* 24, pp. 61-95.

Ciarmita, Massimiliano, and Mark Johnson. (2003). **Supersense tagging of unknown nouns in WordNet**. *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pp.168-175.

Clark, Stephen, and Stephen Pulman. (2007), **Combining Symbolic and Distributional Models of Meaning**. In *Proceedings of the AAAI Spring Symposium on Quantum Interaction*, pp.52-55, Stanford.

Cooper, Robin, Dick Crouch, Jan van Eijck, Chris Fox, Josef van Genabith, Jan Jaspars, Hans Kamp, Manfred Pinkal, Massimo Poesio, Steve Pulman, and Espen Vestre. (1994). **Describing the approaches: FraCaS LRE 62-051**. Deliverable 8, University of Edinburgh.

Cooper, Robin, Dick Crouch, Jan van Eijck, Chris Fox, Josef van Genabith, Jan Jaspars, Hans Kamp, David Milward, Manfred Pinkal, Massimo Poesio, and Steve Pulman. (1996). **Building the framework: FraCaS LRE 62-051**. Deliverable 15, University of Edinburgh.

Copestake, Ann. (1995). **Semantic transfer in Verbmobil**. Verbmobil-report 93, University of Stuttgart and CSLI, Stanford.

Copestake, Ann. (2003). **Report on the design of RMRS**. DeepThought Project Report, http://www.eurice.de/deepthought/downloads_public/D1.1.RMRS.Report.pdf.

Copestake, Ann, Dan Flickinger, Ivan A. Sag, Carl Pollard. (2005). **Minimal Recursion Semantics: an introduction**. *Journal of Research on Language and Computation*, 3(2--3), pp. 281-332.

Crammer, Koby, and Yoram Singer. (2001). **Ultraconservative online algorithms for multiclass problems**. In *Proceedings of the 14th annual Conference on Computational Learning Theory and 5th European Conference on Computational Learning Theory*, pp. 99–115, Amsterdam, The Netherlands.

Cunningham, Hamish. (2000). **Software Architecture for Language Engineering**. Ph.D. Thesis, University of Sheffield. http://gate.ac.uk/sale/thesis.

Curran, James R. (2005). **Supersense tagging of unknown nouns using semantic similarity**. *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, p.26-33.

Curran, James R. and Marc Moens. (2002). **Scaling context space**. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 231–238, Philadelphia, PA, USA.

Dagan, Ido, Shaul Marcus, and Shaul Markovitch. (1995). **Contextual Word Similarity and Estimation from Sparse Data**. *Computer, Speech and Language*, Vol. 9, pp. 123-152.

Dalrymple, Mary, John Lamping, Fernando Pereira, and Vijay Saraswat. (1995). **Linear logic for meaning assembly**. In *Proceedings of CLNLP*.

Davidson, Donald (1967). **The logical form of action sentences**. In Nicholas Rescher, editor, *The Logic of Decision and Action*, University of Pittsburgh Press.

De Boni, Marco, and Suresh Manandhar. (2003). **The Use of Sentence Similarity as a Semantic Relevance Metric for Question Answering**. *New Directions in Question Answering 2003*, pp. 138-144.

de Marneffe, Marie Catherine, and Christopher D. Manning. (2008). **The Stanford typed dependencies representation**. In *COLING Workshop on Cross-framework and Cross-domain Parser Evaluation*, pp. 1-8.

Deese, James E. (1965). *The Structure of Associations in Language and Thought.* Johns Hopkins Press, Baltimore.

Dolan, William B. (1994). **Word Sense Ambiguation: Clustering Related Senses**, *Proceedings of the 15th International Conference on Computational Linguistics*, ACL, Morristown, NJ, pp. 712-716.

DOM. (2004). **Document Object Model Level 3 Core Specification.** A. Le Hors, et al., Editors. World Wide Web Consortium, 13 November 2000. http://www.w3.org/TR/2004/REC-DOM-Level-3-Core.

Dorna, Michael and Martin C. Emele. (1996). **Semantic-based transfer**. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING '96)*, pp. 316-321.

Dorr, Bonnie. (2001). **The LCS Database**. `http://www.umiacs.umd.edu/~bonnie/LCS Database Documentation.html`.

Dörre, Jochen. (1997). **Efficient construction of underspecified semantics under massive ambiguity**. In *Proceedings of ACL/EACL'97*, pp. 386-393.

Elhadad, Michael and Jacques Robin. (1996). **An Overview of SURGE: a Reusable Comprehensive Syntactic Realization Component**. *Technical Report Technical Report 96-03*, Department of Mathematics and Computer Science, Ben Gurion University, Beer Sheva, Israel.

Ervin, Susan. (1961). **Changes with age in the verbal determinants of word-association**. *American Journal of Psychology* 74, pp. 361-372.

Ervin, Susan. (1963). **Correlates of associative frequency**. *Journal of Verbal Learning and Verbal Behavior*, 1(6), pp. 422-431.

Fellbaum, Christiane, editor. (1998). *WordNet: An Electronic Lexical Database*. Bradford Books.

Finch, Steven Paul. (1993). *Finding Structure in Language.* PhD Thesis, University of Edinburgh.

Flickinger, Dan. (2008). **Toward a Cross-Framework Parser Annotation Standard**. In *COLING Workshop on Cross-framework and Cross-domain Parser Evaluation*, pp. 24-28.

Forbes, Katherine, Eleni Miltsakaki, Rashmi Prasad, Anoop Sarkar, Aravind Joshi, and Bonnie Webber. (2001). **D-LTAG System – Discourse Parsing with a Lexicalized Tree Adjoining Grammar**. *Journal of Logic, Language, and Information*, 12, pp. 261-279.

Gale, William A., Kenneth W. Church, and David Yarowsky. (1992). **A Method for Disambiguating Word Senses in a Large Corpus**. *Computers and the Humanities, 26*, pp. 415-439.

Ge, Niyu, John Hale, and Eugene Charniak. (1998). **A statistical approach to anaphora resolution**. *Proceedings of the Sixth Workshop on Very Large Corpora*, pp. 161-171.

Goecke, Daniella and Andreas Witt. (2006). **Exploiting logical document structure for anaphora resolution**. In *Proceedings of LREC 2006 Conference*.

Grefenstette, Gregory. (1992) **SEXTANT: Exploring unexplored contexts for semantic extraction from syntactic analysis**. *Proceedings of the 30th annual meeting of the ACL*, pp. 324-326.

Grefenstette, Gregory. (1993) **Evaluation Techniques for Automatic Semantic Extraction: Comparing Syntactic and Window Based Approaches**. *Workshop on Acquisition of Lexical Knowledge from Text*, SIGLEX/ACL, Columbus, OH.

Grefenstette, Gregory. (1997). SQLET: Short Query Linguistic Expansion Techniques, Palliating One-Word Queries by Providing Intermediate Structure to Text. In *Proceedings of the RIAO'97*, Montreal, Canada, pp. 500-509.

Grefenstette, Gregory. (1999). Light Parsing as Finite-state Filtering. In András Kornai, editor, *Extended Finite State Models of Language*. Cambridge University Press, pp. 86-94.

Grosz, Barbara J. and Candace L. Sidner. (1986). Attention**, intentions, and the structure of discourse**. *Computational Linguistics, 12(3)*, pp. 175-204.

Grosz, Barbara J., Aravind K. Joshi, and Scout Weinstein. (1995). **Centering: A Framework for Modeling the Local Coherence of Discourse**. *Computational Linguistics*, 2(21), pp. 203-225.Hajič, Jan. (1998). **Building a syntactically annotated corpus: The Prague Dependency Treebank**. In Eva Hajičová, editor, *Issues of valency and Meaning. Studies in Honor of Jarmila Panevová*. Karolinum, Prague.

Hajičová, Eva and Ivona Kučerová. (2002). **Argument/valency structure in PropBank, LCS database and Prague Dependency Treebank: A comparative study**. In *Proceedings of LREC*.

Hatzivassiloglou, Vasileios, and Kathleen R. McKeown. (1993). **Towards the automatic identification of adjectival scales: Clustering adjectives according to meaning**. *Proceedings of the 31st Meeting of the Association for Computational Linguistics (ACL '93)*, pp. 172-182.

Hatzivassiloglou, Vasileios, and Kathleen R. McKeown, (1997). **Predicting the semantic orientation of adjectives**. *Proceedings of the 35th Meeting of the Association for Computational Linguistics (ACL '97)*, pp. 174-181.

Hearst, Marti A. (1992). **Automatic acquisition of hyponyms from large text corpora**. *Proceedings of the 14th International Conference on Computational Linguistics*, pp. 539-545.

Hindle, Donald. (1990). **Noun classification from predicate-argument structures**. *Proceedings of the 28th Meeting of the Association for Computational Linguistics (ACL '90)*, pp. 268-275.

Hirschman, Lynette, and Nancy Chinchor. (1997). **MUC-7 Coreference Task Definition**. *Proceedings of the 7th Message Understanding Conference*.

Hobbs, Jerry R. (1977). **Resolving pronoun references**. *Lingua*, Volume 44, pp. 311-338.

Hobbs, Jerry R. (1985). **On the Coherence and Structure of Discourse**. *Technical Report CSLI-85-37*, Stanford University.

Hovy, Eduard H. (1990). **Parsimonious and profligate approaches to the queston of discourse structure relations**. In *Proceedings of the Fifth International Workshop on Natural Language Generation*, Dawson, PA, pp. 128-136.

Hovy, Eduard. (2000). **Language Generation**. In *Encyclopedia of Cognitive Science*, McMillan, London.

Huddleston, Rodney and Geoffrey K. Pullum. (2002). *The Cambridge Grammar of the English Language*. Cambridge University Press, Cambridge.

Johansson, Richard and Pierre Nugues. (2008). **The Effect of Syntactic Representation on Semantic Role Labeling**. In *Proceedings of 22nd International Conference on Computational Linguistic (COLING 22)*, pp. 393-400.

Jurafsky, Daniel and James H. Martin. (2000). *Speech and Language Processing*. Prentice Hall, NJ.

Justeson, John S. and Slava M. Katz. (1991). **Co-occurrences of antonymous adjectives and their contexts**. *Computational Linguistics*, 17 (1).Kaplan, Ronald M., and Joan Bresnan. (1982). **Lexical-Functional Grammar: A Formal System for Grammatical Representation**. In *The Mental Representation of Grammatical Relations*, ed. Joan Bresnan. 173-281. Cambridge, MA: The MIT Press.

Kaplan, Ronald M. and Annie Zaenen. (1989). **Long-distance Dependencies, Constituent Structure, and Functional Uncertainty**. In *Alternative Conceptions of Phrase Structure*, ed. Mark Baltin and Anthony Kroch. Chicago University Press.

Kazai, Gabriella, Mounia Lalmas, Thomas Rölleke. **A model for the representation and focussed retrieval of structured documents based on fuzzy aggregation**. SPIRE 2001: 123-135.

Kendall, Maurice G. (1938). **A new measure of rank correlation**. *Biometrika, 30*, pp. 81-93.

Kennedy, Christopher, and Branimir Boguraev. (1996). **Anaphora for Everyone: Pronominal Anaphora Resolution without a Parser**. *Proceedings of 16th International Conference on Computational Linguistic (COLING 96)* , pp. 113-118.

Kingsbury, Paul and Martha Palmer. (2002). **From TreeBank to PropBank**. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, Las Palmas, Canary Islands, Spain.

Lapata, Mirella, and Chris Brew. (2004). **Verb Class Disambiguation using Informative Priors**. *Computational Linguistics*, 30(1), pp. 45-73

Lappin, Shalom and Herbert J. Leass. (1994). **An Algorithm for Pronominal Anaphora Resolution**. *Computational Linguistics*, 20(4) pp. 535-561.

Leacock, Claudia and Martin Chodorow. (1998). **Combining local context and WordNet similarity for word sense identification**. In C. Fellbaum, editor, *WordNet: An electronic lexical database*, pp. 265-283. MIT Press.

Leacock, Claudia, Martin Chodorow and George A. Miller. (1998) **Using Corpus Statistics and WordNet Relations for Sense Identication**. *Computational Linguistics*, 24(1):147-166.

Lee, Lillian. (1999). **Measures of distributional similarity**. In *Proceedings of ACL '99*, pp. 25–32.

Leech, Geoffrey, Paul Rayson, and Andrew Wilson. (2001). *Word Frequencies in Written and Spoken English: based on the British National Corpus*. Longman, London.

Lenat, Doug B. (1995). **Cyc: A Large-Scale Investment in Knowledge Infrastructure.** *Communications of the ACM*, 38(11), pp. 32-38.

Levy, Roger, and Galen Andrew. (2006). **Tregex and Tsurgeon: tools for querying and manipulating tree data structures**. In *Proceedings of LREC 2006 Conference*.

Li, Hang and Naoki Abe. (1995). **Clustering Words with the MDL Principle**. *Proceedings of the 15th International Conference on Computational Linguistics*, Volume 1, pp. 4-10.

Light, Marc. (1996) **Morphological cues for lexical semantics**. Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics, pp. 25-31.

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

Lin, Dekang. (1998). **Automatic retrieval and clustering of similar words**. In *Proceedings of COLING-ACL 1998*. Montréal, Canada, pp. 768–511.

Lin, Dekang, Shaojun Zhao, Lijuan Qin and Ming Zhou. (2003). **Identifying Synonyms among Distributionally Similar Words**. *Proceedings of IJCAI-03*, pp.1492-1493.

Mann, William C. and Sandra A. Thompson. (1987). **Rhetorical structure theory: A theory of text organization**. *Technical Repoort RS-87-190*, Information Sciences Institute.

Manning, Christopher D. and Hinrich Schütze. (1999). *Foundations of Statistical Natural Language Processing.* The MIT Press, Cambridge, MA.

McCarthy, Joseph F. and Wendy G. Lehnert. (1995). **Using decision trees for coreference resolution**. *Proceedings of the 14th International Conference on Artificial Intelligence*, pp. 1050-1055.

McConville, and Dzikovska. (2008). 'Deep' Grammatical Relations for Semantic Interpretation. In *COLING Workshop on Cross-framework and Cross-domain Parser Evaluation*, pp. 51-58.

McKeown, Kathleen R. (1986). **Text Generation**. *Cambridge University Press*.

Merlo, Paola and Suzanne Stevenson. (2001). **Automatic verb classification based on statistical distributions of argument structure**. *Computational Linguistics* 27, pp. 373 – 408.

Meyers, Adam, Ruth Reeves, and Catherine Macleod. (2004). **NP-External Arguments: A Study of Argument Sharing in English**. In *The ACL 2004 Workshop on Multiword Expressions: Integrating Processing*.

Mihalcea, Rada and Dan Moldovan. (2000). **Semantic indexing using WordNet senses**. In *Proceedings of ACL Workshop on IR & NLP*, Hong Kong.

Miller, George A. and Walter G. Charles. (1991). **Contextual correlates of semantic similarity**. *Language and Cognitive Processes* 6, pp. 1-28.

Mikheev, Andrei. (2000). **Document Centered Approach to Text Normalization**. *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 136 -143.

Mitkov, R., R. Evans, C. Orasan, C. Barbu, L. Jones, and V. Sotirova. (2000) **Coreference and Anaphora: Developing annotating tools, annotated resources and annotation strategies**. In: Baker, P., Hardie, A., McEnery, T. & Siewierska., A. (eds.) *Proceedings of the Discourse Anaphora and Reference Resolution Conference (DAARC2000)*. Lancaster, UK, pp. 49-58.

Miller, George A. (1995). **WordNet: A lexical database**. *Communications of the ACM*, 38(11), pp. 39-41.

Montague, Richard. (1973). **The proper treatment of quantification in ordinary English**. In J. Hintikka, J. Moravcsik, & P. Suppes, editors, *Approaches to Natural Language: Proceedings of the 1970 Stanford Workshop on Grammar and Semantics*, Dordrecht, D. Reidel, pp. 221-242.

Muskens, Reinhard. (1999). **Underspecified semantics**. In Urs Egli and Klaus von Heusinger, editors, *Reference and Anaphoric Relations*, Volume 72 of *Studies in Linguistics and Philosophy*, Kluwer, pp. 311-338.

Ng, Vincent and Claire Cardie. (2001). **Improving machine learning approaches to coreference resolution**. In *Proceedings of the 40th Annual Meeting on Association For Computational Linguistics*, pp. 104-111.

Niehren, Joachim, Manfed Pinkal, and Peter Ruhrberg. (1997). **A uniform approach to underspecification and parallelism**. In Philip R. Cohen and Wolfgang Wahlster, editors, *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, Somerset, New Jersey, pp. 410-417.

Nomoto, Tadashi and Yuji Matsumoto. (1996). **Exploiting Text Structure For Topic Identification**. In *Workshop On Very Large Corpora*, 1996.

Padó, Sebastian, and Mirella Lapata. (2003). **Constructing Semantic Space Models from Parsed Coorpora**. In *Proceedings of the 41st Meeting of the Association of Computational Linguistics*, pp. 128-135.

Pajas, Petr and Jan Štěpánek. (2008). **Recent Advances in a Feature-Rich Framework for Treebank Annotation**. In *Proceedings of 22nd International Conference on Computational Linguistic (COLING 22)*, pp. 673-680.

Palmer, Martha, Rebecca Passonneau, Carl Weir, and Tim Finin. (1993). **The KERNEL text understanding system**. *Artificial Intelligence*, Vol. 63, pp. 17-68.

Parsons, Terence. (1990). Events in the Semantics of English: A Study in Subatomic Semantics. MIT Press, Cambridge, MA.

Pedersen, Ted, and Rebecca Bruce. (1997). **A new supervised learning algorithm for word sense disambiguation**. *Proceedings of the 14th National Conference on Artificial Intelligence*, pp. 604--609.

Pereira, Fernando, Naftali Tishby, and Lillian Lee. (1993). **Distributional clustering of English words**. *Proceedings of the 31st Meeting of the Association for Computational Linguistics (ACL '93)*, pp. 183-190.

Pinkal, Manfred. (1996). **Radical underspecification**. In *Proceedings of the 10th Amsterdam Colloquium*, pp. 587-606.

Pinkal, Manfred. (1999). **On semantic underspecification**. *Computing Meaning*. Kluwer, Dordrecht, pp. 33-55.

Poesio, Massimo. (1996). **Semantic ambiguity and perceived ambiguity**. In Kees van Deemter and Stanley Peters, editors, *Semantic Ambiguity and Underspecification*, CSLI Publications, pp. 159-198.

Polanyi, Livia and Remko Scha. (1984). **A syntactic approach to discourse semantics**, *Proceedings of the 10th International Conference on Computational Linguistics*, Stanford, California, pp. 413-419.

Popescu-Belis, Andrei and Isabelle Robba. (1997). **Cooperation between Pronoun and Reference Resolution for Unrestricted Texts**, *ACL'97 Workshop on Operational Factors in Practical, Robust Anaphora Resolution for Unrestricted Texts*, Madrid, Spain, pp. 94-99.

Priess, Judita. (2002). **Anaphora resolution with memory based learning**. *Proceedings of the 5th UK Special Interest Group for Computational Linguistics (CLUK5)*, pp. 1–8.

Pustejovsky, James. (1991). **The generative lexicon.** *Computational Linguistics* 17(3), pp. 409-441.

Pustejovsky, James, Adam Meyers, Martha Palmer, and Massimo Poesio. (1995). **Merging PropBank, NomBank, TimeBank, Penn Discourse Treebank and Coreference**. *Proceedings of the Workshop on Frontiers in Corpus Annotation II: Pie in the Sky*, pp. 5–12, Ann Arbor.

Resnik, Philip S. (1993). *Selection and Information: A Class-Based Approach to Lexical Relationships.* Ph.D. dissertation, IRCS Report 93-42, University of Pennsylvania, Philadelphia.

Reyle, Uwe. (1993). **Dealing with ambiguities by underspecification: Construction, representation and deduction**. *Journal of Semantics* 10, pp. 123-179.

Richardson, Stephen D., William B. Dolan, and Lucy Vanderwende. (1998). **MindNet: acquiring and structuring semantic information from text**. *Proceedings of the 17th international Conference on Computational Linguistics*, Volume 2, pp. 1098-1102.

Richter, Frank and Manfred Sailer. (1997). **Underspecified Semantics in HPSG**. In Harry Bunt, Leen Kievit, Reinhard Muskens and Margriet Verlinden, editors, *Proceedings of the 2nd International Workshop on Computational Semantics*, pp. 234-246.

Riezler, Stefan, Tracy H. King, Richard Crouch, and Annie Zaenen. (2003). **Statistical Sentence Condensation using Ambiguity Packing and Stochastic Disambiguation Methods for Lexical-Functional Grammar**. *Proceedings of HLT-NAACL 2003 Main Papers*, pp. 118-125, Edmonton.

Ruppenhofer, Josef, Michael Ellsworth, Miriam R. L. Petruck, and Christopher R. Johnson. (2005). **FrameNet II: Extended Theory and Practice**. *ICSI Technical Report*.

Sag, Ivan A., Timothy Baldwin, Francis Bond, Ann Copestake and Dan Flickinger. (2002). **Multiword Expressions: A Pain in the Neck for NLP**. In Alexander Gelbukh, editor, *Proceedings of CICLING-2002*, Springer.

Saint-Dizier, Patrick and Evelyne Viegas, editors. (1995). *Computational Lexical Semantics*. Cambridge University Press.

Schiehlen, Michael. (1996). **Semantic construction from parse forests**. *Proceedings of the 16th International Conference on Computational Linguistics (COLING '96)*.

Schiehlen, Michael. (1997). **Disambiguation of Underspecified Discourse Representation Structures under Anaphoric Constraints**. *Proceedings of the 2nd International Workshop on Computational Semantics*.

Schnieder, Gerold (1998). **A Linguistic Comparison of Constituency, Dependency and Link Grammar**. *ExtrAns Research Report: Dependency vs. Constituency*. Diploma Paper, Universität Zürich.

Schuler, Karin K. (2005). **Verbnet: a broad-coverage, comprehensive verb lexicon.** *Doctoral Thesis. UMI Order Number: AAI3179808*, University of Pennsylvania.Schütze, Hinrich. (1995). **Distributional part-of-speech tagging**. Proceedings of the 7th Meeting of the European Chapter of the Association for Computational Linguistics (EACL 7), pp. 141-148.

Schütze, Hinrich. (1998). **Automatic word sense discrimination**. *Computational Linguistics* 24, pp. 97-124.

Setzer, Andrea. (2001). **Temporal Information in Newswire Articles: an Annotation Scheme and Corpus Study**, *Unpublished PhD thesis*, University of Sheffield.

Sgall, Petr, Jarmila Panevova, and Eva Hajičová. (2004). **Deep Syntactic Annotation: Tectogrammatical Representation and Beyond**. *HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation*.

Siddharthan, Advaith. (2003). **Resolving Pronouns Robustly: Plumbing the Depths of Shallowness**. In *Proceedings of the Workshop on Computational Treatments of Anaphora*,

11th Conference of the European Chapter of the Association for Computational Linguistics (EACL'03), Budapest, pp. 7-14.

Sleator, Daniel, and Davy Temperley. (1993). **Parsing English with a Link Grammar**. *Third International Workshop on Parsing Technologies*.

Soon, Wee Meng, Hwee Tou Ng, and Daniel Chung Yong Lim. (2001). **A machine learning approach to coreference resolution of noun phrases**. *Computational Linguistics* 27(4), pp. 521–544.

Srinivas, Bangalore, Christine Doran, Beth Ann Hockey, and Aravind Joshi. (1996). **An approach to robust partial parsing and evaluation metrics**. In *Proceedings of the Workshop on Robust Parsing at European Summer School in Logic, Language and Information*, Prague.

Srinivas, Bangalore. (2000). **A lightweight dependency analyzer for partial parsing**. *Natural Language Engineering*, 6(2), pp. 113-138.

Tateisi, Yuka. (2008). **Toward an Underspecifiable Corpus Annotation Scheme**. In *COLING Workshop on Cross-framework and Cross-domain Parser Evaluation*, pp. 17-23.

Torrent, Gemma, and Laura Alemany. (2003). **Clustering Adjectives for Class Acquisition**. *Proceedings of the 10th Conference on European Chapter of the Association for Computational Linguistics*, Vol 2, Student Research Workshop Session, pp. 9-16.

Trujillo, Arturo. (1995). **Lexicalist Machine Translation of Spatial Prepositions**. Ph.D. Thesis, Cambridge University.

van Deemter, Kees. (1996). **Towards a logic of ambiguous expressions**. In Kees van Deemter and Stanley Peters, editors, *Semantic Ambiguity and Underspecification*, CSLI Publications, pages 203-235.

van Deemter, Kees and Stanley Peters, editors. (1996). **Semantic Ambiguity and Underspecification**. CSLI Publications.

Vilain, Marc, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. (1995). **A Model-theoretic Coreference Scoring Scheme**. *Proceedings of the 6th Message Understanding Conference (MUC6)*, Morgan Kaufmann, San Mateo, CA, pp. 45-52.

Voorhees, Ellen M. and Dawn M. Tice. (2000). **The TREC-8 question answering track evaluation**. In Ellen M. Voorhees and Dawn M. Tice, editors, *Proceedings of the 8th Text Retrieval Conference (TREC-8)*, NIST Special Publication 500-246, pp. 83-105.

Walker, Marilyn A. (1998). **Centering, Anaphora Resolution, and Discourse Structure**. In Marilyn A. Walker, Aravind K. Joshi, and Ellen F. Prince, editors, *Centering in Discourse*. Oxford University Press.

Waterman, Scott. (1996). **Distinguished Usage**. In Branimir Boguraev and James Pustejovsky, editors, *Corpus Processing for Domain Acquisition*, MIT Press, Cambridge, MA, pp. 143-172.

Wilcock, Graham. (2001). **Pipelines, Templates, and Transformations: XML for Natural Language Generation**, *Proceedings of the 1st NLP and XML Workshop*, pp.1-8.

Woods, William A. (1975). **What's in a Link: Foundations for Semantic Networks**, in D.G. Bobrow & A. Collins, editors, *Representation and Understanding*, Academic Press, New York, pp.35-82.

Yarowsky, David. (1995). **Unsupervised word sense disambiguation rivalling supervised methods**. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pp. 189-196.

Yeh, Alexander. (2000). **Using existing systems to supplement small amounts of annotated grammatical relations training data**, *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, Hong Kong, pp.126-132.

Appendix A

The following table lists the output of the CAMEO framework on the sentences reported in Canning (2003). For each sentence the table gives the original passive sentence (numbered), the output from SYSTAR (S) as reported in Canning, and the output of CAMEO (C). (Note the syntactic parses used in SYSTAR were not available. Parse errors were hand-corrected for the CAMEO representation. See Section 4.2.2) Sentences which were judged incorrect grammatically or semantically (in relation to the original sentence) are shown with a (*).

In a few cases a second passive phrase was observed that was not activised in the SYSTAR output. The activisation of both phrases is shown for the CAMEO output (C2).

1. She is impressed by the changes in the city , particularly the proposed introduction of the Metro .

S * The changes in the city impressed her particularly the proposed introduction of the Metro.

C The changes in the city impress her , particularly the proposed introduction of the Metro .

2. Alan , who is sponsored by Washington-based outdoor clothing and equipment manufacturer Berghaus , has now reached the summit of Makalu , the fifth highest peak in the world .

S * Washington-based outdoor clothing and equipment manufacturer Berghaus has now reached the summit of Makalu, the fifth highest peak in the world sponsored Alan, who.

C Alan , who washington-based outdoor clothing equipment manufacturer Berghaus sponsors , has now reached the summit of Makalu , the fifth highest peak in the world .

3. Emma Rae , 15 , of Parkhurst Road , said her three-year-old brother , James , suffered a broken leg when he was knocked down by a car on Sunday , not far from where the little girl died .

S Emma Rae, 15, of Parkhurst Road said her three-year-old brother, James suffered a broken leg when a car on Sunday, not far from where the little girl died knocked him down.

C Emma Rae , 15 , of Parkhurst road , said her three-year-old brother , James , suffered a broken leg when a car on Sunday knocked him down , not far from where the little girl died .

4. When demonstrators returned this morning they were joined by the consultant who treated the dead girl .

S When demonstrators returned this morning the consultant joined them who treated the dead girl.

C When demonstrators returned this morning , the consultant , who treated the dead girl , joined them .

5. She says she was told by her doctor that it related to the batch of vaccine with which her son was injected .

S She says her doctor told her that it related to the batch of vaccine with which her son was injected.

C She says her doctor told her that it related to the batch of vaccine , with which her son was injected .

6. Today the report prompted local parents who suspect their children were harmed by MMR - including one couple who won $30,000 in Government damages - to speak out .

S * MMR harmed today, the report prompted local parents who suspect their children including one couple who won £30,000 in Government damages - to speak out.

C Today the report prompted local parents , who suspect MMR harmed their children including one couple , who won $30,000 in government damages , to speak out .

7. Joan Gray lost her husband John , who was among nine crew killed when the 25,000 tonne tanker burst into flames after it was hit by the Panamanian-registered Western Winner in fog off the Belgian Port of Ostend .

S * The Panamanian-registered Western Winner in fog off the Belgian Port of Ostend hit Joan Gray lost her husband John, who was among nine crew killed when the 25,000-tonne tanker burst into flames after it.

C Joan Gray lost her husband John who was among nine crew killed when the 25,000 tonne tanker burst into flames after the panamanian-registered western winner hit it in fog off the belgian port of Ostend .

8. He was struck down by the brain disease last October .

S The brain disease last October struck him down.

C The brain disease struck him down last october .

9. The injured animals were discovered by Paul Barrow when he went to feed them at his father 's small holding between Melrose Crescent and Ambleside Avenue .

S Paul Barrow when he went to feed them at his father's small holding between Melrose Crescent and Ambleside discovered the injured animals.

C Paul Barrow discovered the injured animals when he went to feed them at his father's small holding between Melrose and crescent Ambleside Avenue .

10. Last year the campaign was supported by 38 primary schools with a further five joining in this time .

S 38 primary schools with a further five joining in this time supported last year the campaign.

C Last year 38 primary schools supported the campaign with a further five joining in this time .

11. The tenants were angered by a letter from Vaux giving details of price rises which will take effect after the brewery closure .

S A letter from Vaux giving details of price rises which will take effect after the brewery closure angered the tenants.

C A letter from Vaux giving details of price rises , which will take effect after the brewery closure angered the tenants .

12. Frank Nicholson , managing director of Vaux Breweries , fought hard to save the brewery but bids from his management buy-out team were rejected by the Swallow Group .

S * The Swallow Group rejected Frank Nicholson, managing director of Vaux Breweries fought hard to save the brewery but bids from his management buy-out team.

C Frank Nicholson managing director of Vaux Breweries fought hard to save the brewery , but the Swallow Group rejected bids from his management buy-out team .

13. They were disturbed by a neighbour .

C A neighbour disturbed them .

S A neighbour disturbed them.

14. Mr Clifford , a single man who is now on police bail , was informed by post .

S Post informed Mr Clifford, a single man who is now on police bail.

C Post informed Mr Clifford a single man , who is now on police bail , .

15. The campaign was kicked off by Sunderland AFC mascots Samson and Delilah at St Godric 's RC Primary School , Durham City .

S* Sunderland AFC mascots Samson and Delilah at St Godric's RC Primary School kicked the campaign Durham City off.

C Sunderland AFC mascots Samson and Delilah at St Godric's RC Primary School , Durham City kicked off the campaign .

16. Two years later the premises in High Street West were destroyed by fire and a building was constructed on the John Street site , opening in May 1956 .

S * Fire and a building was constructed on the John Street site, opening in May 1956 destroyed two years later the premises in High Street West.

C Two years later fire destroyed the premises in High Street West and a building was constructed on the John Street site opening in may 1956 .

17. Tourist chiefs in the North East were disappointed by the results which they believe could have been affected by bad weather .

S The results which they believe could have been affected by bad weather disappointed tourist chiefs in the North East.

C The results , which they believe could have been affected by bad weather disappointed tourist chiefs in the north east .

C2 The results , which they believe bad weather could have affected disappointed tourist chiefs in the north east .

18. The city was cited by the Joseph Rowntree Foundation as an example of what can be achieved in areas that suffer job losses after the collapse of traditional heavy industries .

S* The Joseph Rowntree Foundation as an example of what can be achieved in areas that suffer job losses after the collapse of traditional heavy industries cited the city.

C The Joseph Rowntree foundation cited the city as an example of can achieve what in areas , that suffer job losses after the collapse of traditional heavy industries .

C2 The Joseph Rowntree foundation cited the city as an example of what can be achieved in areas , that suffer job losses after the collapse of traditional heavy industries .

19. Mafeking Street , on Ford Estate , takes its name from the town in northern South Africa which was defended by the British against the Boers at the outbreak of the Boer War in 1899 .

S* Mafeking Street, on Ford Estate takes its name from the town in northern South Africa the British against the Boers as the outbreak of the Boer war in 1899 defended which.

C Mafeking Street , on Ford estate , takes its name from the town in northern South Africa , which the British defended against the Boers at the outbreak of the Boer war in 1899 .

20. A 34-year-old Bristol man suffered a broken leg and pelvic injuries after he too was trapped by the bus .

C A 34-year-old Bristol man suffered a broken leg and pelvic injuries , after the bus trapped him too .

S A 34-year-old Bristol man suffered a broken leg and pelvic injuries after the bus trapped him too.

21. Those inquiries were launched by Transport Minister John Reid , at the request of Sunderland 's two MPs , after an Echo investigation uncovered concerns that automatic buses may have been affected by uncontrollable power surges since the 1980s .

S Transport Minister John Reid launched those inquiries at the request of Sunderland's two MPs, after an Echo investigation uncovered concerns that automatic buses may have been affected by uncontrollable power surges since the 1980s.

C Transport Minister John Reid launched those inquiries at the request of Sunderland's two MPs after an echo investigation uncovered concerns that uncontrollable power surges since the 1980s may have affected automatic busses .

C2 Transport Minister John Reid launched those inquiries at the request of Sunderland's two MPs after an echo investigation uncovered concerns that automatic busses may have been affected by uncontrollable power surges since the 1980s .

22. But the council feels some motorists would ignore the slower speed unless this was backed up by traffic calming measures .

S But the council feels some motorists would ignore the slower speed unless traffic calming measures backed this up.

C But the council feels some motorists would ignore the slower speed unless traffic calming measures backed this up .

23. Stephanie Cook , three , of Pennycross Road , Pennywell died when she was knocked down by a car on Hylton Road last week .

S Stephanie Cook, three, of Pennycross Road, Pennywell died when a car on Hylton Road last week knocked her down.

C Stephanie Cook three of Pennycross road Pennywell died when a car knocked her down on Hylton road last week .

24. Glass in the two remaining stands has been replaced by wire mesh .

C Wire mesh has replaced glass in the two remaining stands .

S Wire mesh has replaced glass in the two remaining stands.

25. The fact that the jobless total of about 82,600 has not risen has been welcomed by business chiefs .

C Business chiefs have welcomed the fact , that the jobless total of about 82,600 has not risen , .

S Business chiefs have welcomed the fact that the jobless total of about 82,600 has not risen.

26. In the last year , £66.4 million has been saved by anti-fraud officers working on investigations across the North East , said fraud manager Chris Mason .

C In the last year , anti-fraud officers working on investigations across the north east have saved £66.4 million said fraud manager Chris Mason .

S Anti-fraud officers working on investigations across the North East, said fraud manager Chris Mason have saved in the last year, £66.4 million.

27. Mr Donkin 's anger has been fuelled by a letter written by Mr Walls to the council 's Conservative leader Coun Margaret Forbes .

S A letter written by Mr Walls to the council's Conservative leader Coun Margaret Forbes has fuelled Mr Donkin's anger.

C A letter Mr Walls wrote to the council's conservative leader Coun Margaret Forbes has fuelled Mr Donkin's anger .

C2 A letter written by Mr Walls to the council's conservative leader Coun Margaret Forbes has fuelled Mr Donkin's anger .

28. A giant $ 25million leisure plan for Sunderland city centre is being considered by councillors tonight .

S Councillors tonight are considering a giant £25million leisure plan for Sunderland city centre.

C Councillors are considering a giant £ 25million leisure plan for Sunderland city centre tonight .

29. Sunderland 's critical shortage of doctors could be eased by a pioneering new pilot scheme expected to be given the go-ahead by health chiefs tomorrow .

S A pioneering new pilot scheme expected to be given the go-ahead by health chiefs tomorrow could ease Sunderland's critical shortage of doctors.

C A pioneering new pilot scheme expected to be given the go-ahead by health chiefs tomorrow could ease Sunderland's critical shortage of doctors .

C2 A pioneering new pilot scheme expected health chiefs to give the go-ahead tomorrow could ease Sunderland's critical shortage of doctors .

30. A signed safc shirt and ball are to be auctioned by the James Herriot visitor centre .

S The James Herriot visitor centre is to auction a signed safc shirt and ball.

C The James Herriot visitor centre is to auction a signed safc shirt and ball .

31. A cramped Sunderland school is to be replaced by a new $ 1million development after winning Government cash .

S A new £1million development after winning Government cash is to replace a cramped Sunderland school.

C A new £ 1million development is to replace a cramped Sunderland school after wining government cash .