Frame-based semantic role labeling

Master thesis
Language and Speech Technology

J.H.J. Gubbels
Utrecht University
Supervisor: Prof. dr. M. Moortgat

January 24, 2007

Contents

Preface

The composition of this thesis was originally intended to be a three-month project aiming to combine semantic web techniques with modern natural language processing. Due to the common problem of initial lack of focus, the project started out as an expedition into many fields surrounding the two topics. The ultimate challenge was to find a topic in which semantically motivated knowledge bases, similar to the ontologies used on the semantic web, are adopted to construct semantic representations of natural language utterances. The search for knowledge bases that combine semantic knowledge on concepts with information on the syntactic realization of these concepts in natural language, narrowed the possibilities for research. In fact all found knowledge bases that live up to these criteria, were created to aid the automated detection of *semantic roles*. In semantic roles the topic of the thesis was found and the result was an attempt to build an automatic semantic role labeler based on one of the found knowledge bases. The thesis should present the reader with a good overview of recent research on semantic role labeling and with a nice example of what this research can lead to in practice.

I would like to take the opportunity of this preface to acknowledge my gratitude to some of my beloved ones. First, I would like to thank my partner Annette, a brilliant mathematician and astonishing mother, but also a keen editor and intellectual sparing partner, for being able to encourage and withstand me at the same time. Secondly, I owe gratitude to my two inspiring children Rafael and Ella. Both were born during the writing of this thesis. At difficult times of crashing computers and failing algorithms their sheer presence forced me to pick up pieces quickly and uphold my character as a well-balanced father. Third I need to thank my good friends Willem, Sander and Luite. I hope their constructive criticism and understanding will be as warm regarding upcoming issues in my life, as they were regarding my thesis proceedings. All these people and many more need to be mostly thanked for never seeming to have lost faith in my academic competences.

A special word of thanks should finally be addressed to my supervisor Prof. Dr. Michael Moortgat, who was fair and to-the-point at the scarce moments of me actually delivering revisable material, while remaining patient and understanding.

Chapter 1

Introduction

1.1 Semantic role labeling

The assessment that now it is time to start applying well-known techniques for formal and computational semantics to a larger scale, and get detailed semantic analysis from wide-coverage parsers, is often heard among computational linguists [Bos 2005, Blackburn 2005, Gildea and Palmer 2002]. The Natural Language Processing (NLP) community has produced sophisticated parsers, achieving high coverage and producing accurate syntactic analysis. Having somehow mastered the artifact of syntax, it seems that computational linguistics can finally start to deal with the real focus of interest: linguistic meaning.

While the NLP community has started investigating challenging areas in computational semantics such as anaphora resolution, resolving ambiguity and scope construel, this thesis has more modest intentions. The thesis proposes an algorithm that identifies and labels constituents of English sentences that fill *semantic roles* evoked by a target *predicator* (e.g. a verb, noun or adjective).

A semantic role is the actual role a component (animate or inanimate participant) plays in some real or imagined situation, apart from the linguistic encoding of those situations [Payne 1997]. The notion of semantic role is also known as *semantic case*, *thematic role*, *theta role* (in generative grammar), and *deep case* (in case grammar).

In the context of the mentioned task, a predicator denotes a predicate that describes the relation among different components in a situation or event. These participants can be considered the arguments of the predicate. The minimum number of arguments of a predicator is zero: Certain verbs indicating weather conditions such as *rain* and *snow* have no depending semantic roles. On the other hand there are examples, such as 1.1, in which a verb can take up to four arguments or more.

[John] **leased** [the apartment] [to Bill] [for 1000$ a month]. (1.1)

Automated recognition of predicate-argument structures is made difficult by the multiple ways in which the same predicating target, evoking the same semantic roles, can be realized at syntactic level. The sentences in 1.2 and 1.3 for example, have different syntactic structures but describe the same event with the same components.

[John] **will meet** [Mary]. (1.2)
[John] **will meet** [with Mary].
[John] and [Mary] **will meet**.

[The door] **opened**. (1.3)
[John] **opened** [the door].

Task of the proposed algorithm is to correctly identify and label the participants in an event, no matter how they are syntactically expressed.

1.2 Practical relevance

Being able to annotate natural language with semantic roles would be of great use to a wide variety of linguistic research. Corpora with semantic role annotation offer rich data both to empirical investigations in lexical semantics and large-scale lexical acquisition for NLP applications [Ellsworth et al. 2005].

Correctly identifying the semantic roles of sentence constituents is a crucial part of interpreting text, and in addition to forming an important part of the information extraction problem, can serve as an intermediate step in machine translation or automatic summarizing [Gildea and Palmer 2002].

The shallow level of semantic interpretation of predicate-argument structure specified by semantic roles, can be put to use for word-sense disambiguation, where the roles associated with a word can be cues to its sense. The semantic representation can further be used by knowledge management systems to perform a semantic analysis similar to the one the Semantic Web community proposes. Also, incorporating semantic roles into probabilistic models of language may eventually yield more accurate parsers and better language models for speech recognition. [Gildea and Jurafsky 2002].

A concrete and simple example of the use of semantic roles in information retrieval is given in [Giuglea and Moschitti 2004] and considers a regular Internet search engine query trying to retrieve information about '*teenagers arrested*'. Without a sense of semantic role information, this query will return all documents in which teenagers appear in the context of an arrest even if they are not the offender, or worse, even if they have nothing in common with the offense. The same happens when trying to find out '*who was arrested for robbing the jeweler in Fernley Shopping Centre?*' in a question-answering system. In fact this case is even more complicated as the complexity of the question increases with its length. Now, if semantic role information would be available, it would be possible to look for a *Suspect* that committed a specific *Offense* in a particular situation. This would arguably result in a noticeable increase of the accuracy of the answer.

1.3 Scientific relevance

This thesis is strongly related to a field of research that emerged at the end of the 20th century. Due to the availability of large syntactically annotated cor-

pora and sophisticated systems to aid detection of predicate-argument structure, it became possible to automatically label sentence constituents with semantic roles.

Many recent information extraction systems for limited domains rely on finite-state systems that do not construct a full syntactic derivation for the sentence being analyzed. Some of these systems consist of finite-state recognizers for various entities, which are cascaded to form recognizers for higher-level relations [Hobbs et al. 1997]. Others use low-level 'chunks' from a general-purpose syntactic analyzer as observations in a trained Hidden Markov Model [Ray and Craven 2001].

Such approaches have a large advantage in speed, as the extensive search space that modern statistical parsers have to deal with, is avoided. Nevertheless it is often expected that the attachment decisions made by a parser are relevant to determine whether a constituent of a sentence denotes an argument of a particular predicate, and what its relation to the predicate is.

In [Gildea and Palmer 2002] a system to predict semantic roles from sentences and their parse trees as determined by a statistical parser [Collins 1997] is compared with a chunk-based system[1] [Tjong Kim Sang and Buchholz 2000]. First conclusion was that the latter performs worse than the former and that statistical parsers, although computationally expensive, do a good job of providing relevant information for semantic interpretation. Second conclusion is that parsers still have a long way to go and that improvements in parsing will translate directly into better semantic representations.

Table 1.1 sums more recent research that is of particular relevance to this thesis and the preliminaries on which this research is based. In short, the mentioned literature in the right column of table 1.1 reports results of performance testing for different automated semantic role labeling approaches. These approaches can be distinguished through two main points of difference:

1. The type of semantic classification of the predicators that are target of the labeling process;

2. The parsing model that is used for identifying the sentence constituents that might fill semantic roles.

The first point is elaborately discussed in sections 2.2 and 2.3, the second is subject to sections 3.1 and 3.2.

Of course recent research on automated semantic role labeling is not restricted to English. It is worth to mention the Dutch Language Corpus Initiative (D-Coi), which focuses among others on semantic role assignment for Dutch Corpora [Monachesi and Schuurman 2006]. Further the Chinese Language Processing program at Penn University (US), part of the Automatic Content Extraction (ACE) program, evokes simular research efforts for a Chinese corpus[Xue and Palmer 2005]. The program aims at the construction of a Chinese Proposition Bank, a corpus of Chinese text annotated with information about basic semantic propositions.

[1] More precisely: the chunks are derived from a large collection parse trees using the *conversion* described in [Tjong Kim Sang and Buchholz 2000]. Thus, the experiments were carried out using 'gold standard' rather than automatically derived chunk boundaries, which arguably provides an upper bound on the performance of a chunk-based system.

Levin's classification	
Statistical Treebank Parser [Collins 1997]	[Gildea and Hockenmaier 2003]
Statistical CCG Parser [Hockenmaier and Steedman 2002]	[Gildea and Hockenmaier 2003]
Frame Semantics	
Statistical Treebank Parser [Collins 1997]	[Gildea and Jurafsky 2002]
Statistical CCG Parser [Hockenmaier and Steedman 2002]	

Table 1.1: Recent research regarding automated semantic role labeling. CCG abbreviates *Combinatory Categorial Grammar* [Steedman 2000].

1.4 Research question

The algorithm presented in part two of this thesis makes use of a linguistic resource that is based on frame semantics and considers predicating *verbs*, *nouns*, *adverbs* and *prepositions*. This sets the algorithm apart from the research in [Gildea and Hockenmaier 2003], which adopts a resource based on Levin's verb classification and does therefor not consider predicators other than verbs. In [Gildea and Jurafsky 2002] predicators are classified based on frame semantics, however, both the algorithm for semantic role labeling and the parser of choice differ from those proposed in this thesis. The latter gives rise to the research question central to the second part of the thesis:

> *Can the use of a statistical CCG parser combined with a rule-based labeling algorithm lead to higher coverage and accuracy on a common labeling task than the use of the tree parser and a statistical labeling algorithm as proposed in [Gildea and Jurafsky 2002]? And why (not)?*

A description of the exact differences between the two approaches and thus the nature of the comparison has to wait until part two of this thesis. First, part one provides context to the main issues of automated semantic role assignment.

Part I

Theory

Chapter 2

Levin's Classification versus Frame Semantics

2.1 Premises of predicator classification

The previous chapter mentioned the existence of different semantic classifications of predicators. The algorithm for semantic role labeling proposed in part two this thesis adopts a lexical resource that adheres to so-called frame semantics. The use of this purely semantically motivated predicator classification is, perhaps surprisingly, relatively new to computational linguistics. Most research on semantic role labeling makes use of predicator classifications that are, fundamentally, syntactically motivated. In order to provide the reader with the necessary background knowledge, this chapter simplifies the issue of syntactic versus semantic premises of predicator classification, to a discussion of Levin's classification versus frame semantics respectively. Given the difference, existing linguistic resources that might aid automated semantic role labeling will be put forward and the adoption of a semantically motivated classification in this thesis will be explained.

2.2 Levin classes versus frames

In most of the research presented in chapter one, the notion of semantic role is determined by the adoption of *Levin's verb classification* [Levin 1993]. In this classification verbs are grouped together based on their syntactic behavior. The resulting *Levin classes* are coherent from a semantic point of view, as all verbs in such a class share, by definition, the same semantic roles.

Levin classes are formed according to a large set of *diathesis alternation* criteria. Diathesis alternations are variations in the way verbal-arguments are grammatically expressed consistently with a specific semantic phenomenon. Consider the examples of two different types of diathesis alternation in 2.1 and 2.2.

Middle Alternation: (2.1)
[The butcher $_{Subject,Agent}$] cuts [the meat $_{Direct\ Object,Patient}$].
[The meat $_{Subject,Patient}$] cuts easily.

Causative/inchoative Alternation: (2.2)
[Janet $_{Subject,Agent}$] broke [the cup $_{Direct\ Object,Patient}$].
[The cup $_{Subject,Patient}$] broke.

In both cases what is alternating is the grammatical function that the *Patient* role takes when changing from the transitive use of the verb to the intransitive use. More precisely the semantic role of the subject of the intransitive use of the verb is the same as the semantic role of the direct object of the transitive use. If the two mentioned alternation criteria were the only ones that existed, the syntactic behavior of *cut* and *break* in these alternations, would cause them to be members of the same Levin class.

Levin builds her classification starting from the assumption that there is a strong connection between syntax and semantics. This connection can be exploited by annotating arguments with semantic roles at diathesis level: For verbs pertaining to the same Levin class the semantic roles will be the same. From the perspective of an annotation task this is useful as the identification of the predicate-argument structure by a sophisticated parser combined with knowledge on the Levin class are sufficient to predict the semantic roles filled by sentence constituents.

The use of Levin's classification nevertheless has some important down-sides. The first that needs to be mentioned considers the low granularity of the classification. Levin's list of 193 verb classes is divided into 51 sections, with two further levels of subdivision. The sections reflect a limited attempt to group verb classes related by meaning together. However, there is little hierarchical organization compared to the number of classes identified [Levin 1993]. Because roles labels are defined per verb class, the classification of verb classes into higher-level classes is not trivial.

Secondly, although the whole thesis of Levin's work is that grouping words according to alternations tends to produce semantically coherent classes, it can also split words that are close in meaning, or lump semantically disparate words together [Baker and Ruppenhofer 2002]. Alternations dictate for example that *fill* and *load* be in a separate class. These could however also be considered idiosyncrasies belonging to the same semantic grouping.

Finally, Levin classifies only verbal predicators. There is however much evidence that also other parts-of-speech can act as predicator. Consider for example the noun *argument* in the sentence *I witnessed a heated argument among the players over a game of cards.* In this sentence *argument* could be considered evoking semantic roles *Arguers* (the players) and *Issue* (a game of cards). The role of diathesis alternations regarding noun phrases are less clear with nouns as they are with verbs.

The mentioned downsides are of great importance when considering the value of the semantic representations that are build when specifying predicate argument structure through semantic roles. From the perspective of interpretation of these representations, i.e. their use in reasoning tasks, it is of great value to have a semantically consistent and fine-grained classification of predicators and their arguments.

An example: Since alternation patterns are critical in Levin's system, alternators and non-alternators cannot be in the same verb class. This means the interchangeability between the verbs *fill* and *load* in 2.3 is not captured as *fill*

is subject to causative/inchoative alternation (2.4), while *load* is not (2.5).

John loaded the barrow with paving stones. (2.3)

Mary waited as the barrow filled with paving stones. (2.4)

*Mary waited as the barrow loaded with paving stones. (2.5)

This would mean that this classification would not be supportive to a 'reasoning task' to determine the answer to the question *Who **filled** the barrow with paving stones?* through analysis of 2.3. Reasoning should yield that the *Agent*s of a *Fill* and a *Load* event are the same if the 'events' are equivalent.

In order to overcome the problems with Levin's classification a more semantically motivated approach to verb classification needs to be pursued. An approach in which verbs are grouped according to the conceptual structures that underlie them. This means that verbs might be grouped together that are semantically similar but have different (or no) alternations, and that verbs which share the same alternation might be represented in two different semantic frames.

A theory that classifies predicates by the concepts which they express is called *frame semantics*. In frame semantics a predicator activates, or evokes, a *frame* of semantic knowledge relating to the specific concept it refers to. The basic idea is that one cannot understand the meaning of a single word without access to all the essential knowledge that relates to that word. For example, one would not be able to understand the word *sell* without knowing anything about the situation of *Commercial_Transfer*, which involves, among other things, a *Seller*, a *Buyer*, *Goods*, *Money* and so on. The participants in such a situation are called *frame elements* and can be considered semantic role fillers.

Defining semantic roles at this intermediate frame level allows some generalization across the roles of different predicators. Predicators add additional semantics to a general frame, or highlight a particular aspect of that frame. Also several types of relations between frames can be defined. The most common are:

- *Inheritance*: The child frame is a subtype of the parent frame, and each frame element in the parent is bound to a corresponding frame element in the child. An example might be the *Revenge* frame which inherits from the *Rewards_and_punishments* frame;
- *Use*: The child frame presupposes the parent frame as background, e.g. the *Speed* frame might 'use' (or presuppose) the *Motion* frame. Here not all parent frame elements need to be bound to child frame elements;
- *Sub-frame*: The child frame is a sub-event of a complex event represented by the parent, e.g. the *Criminal_process* frame might have sub-frames *Arrest*, *Arraignment*, *Trial*, and *Sentencing*.

It is not hard to imagine how these relations would be of great benefit to reasoning tasks such as the one presented in the load/fill example above.

One way of thinking about traditional abstract semantic roles, such as *Agent* and *Patient*, in the context of frame semantics, is that they are frame elements defined by abstract frames such as *Action* and *Motion*, at the top of an inheritance hierarchy of semantic frames [Fillmore and Baker 2000].

2.3 Corpora annotated with semantic roles

Any approach to wide-coverage automated semantic role labeling will depend on some large resource that records variations in syntactic realization of predicate-argument structure. The distinction between Levin's classification and frame semantics can be used to classify the resources that are available for this purpose. In the short list below the wide variety of taxonomies, ontologies, knowledge bases, lexicons and thesauri, such as SUMO, WordNet, FreeNet, ConceptNet etc., aiming at representing lexical concepts underlying words are left out. Only those corpora and lexicons that are explicitly documenting the range of semantic and syntactic combinatory possibilities (valences) of each word in each of its senses, are of interest here.

- *VerbNet* [Kipper et al. 2000], a lexicon that implements a refinement of Levin's classification. Levin's verb classes are used to systematically construct lexical entries;
- *PropBank* [Kingsbury et al. 2002], the Penn TreeBank [Marcus et al. 1993] annotated with the predicate-argument structure of occurring verbs. The Penn TreeBank is a corpus of linguistic parse trees of sentences stemming from the Wall Street Journal. The annotation in PropBank is based on the verb classes defined in the VerbNet lexicon;
- *FrameNet* [Ruppenhofer et al. 2005], a full lexical resource based on frame semantics and supported by corpus evidence.

VerbNet only records a specific part of the syntactic realization of predicate-argument structure. Besides the actual classification of verbs, VerbNet contains small sentences that illustrate the different diathesis alternations of predicating verbs belonging to a particular verb class. Consider for example the benefactive alternations for the verb *carve*, which are recorded in VerbNet as depicted in table 2.1. The second line in each alternation indicates the predicator (indicated by a V) and the chunks of the sentence that fill the class specific semantic roles. The precise syntactic relation between these chunks or the words forming them is not provided.

Mapping VerbNet alternation patterns to the derivation trees in the Penn TreeBank has produced PropBank, i.e. the Penn TreeBank corpus annotated with VerbNet's semantic roles. This way PropBank extends VerbNet with a corpus of examples of syntactic relations between VerbNet semantic roles and predicators.

The Berkeley FrameNet project is committed to defining rich semantic frames on which annotation is based. FrameNet is less concerned with annotating complete texts, concentrating instead on annotating a set of examples of each syntactic syntactic realization for each predicator (including verbs, nouns and adjectives), and attempting to describe the network of relations among the semantic frames through an ontology. In FrameNet, predicators belong to frames based on a shared semantics. They need not exhibit all the same syntactic behaviors in order to be grouped together. Moreover FrameNet does not only define inheritance relations between frames, but also has a parallel, detailed set of inheritance relationships among the frame elements: the semantic roles.

PropBank and FrameNet have different aims, which are reflected in their design. The practical aim of PropBank was to obtain a complete semantic role

Benefactive Alternation (double object) *Martha carved the baby a toy out of a piece of wood* Agent V Beneficiary Product Prep(from out of) Material
Benefactive Alternation (for variant) *Martha carved a piece of wood for the baby* Agent V Material Prep(for) Beneficiary
Benefactive Alternation (for variant) *Martha carved a piece of wood into a toy for the baby* Agent V Material Prep(into) Product Prep(for) Beneficiary
Benefactive Alternation (for variant) *Martha carved a toy for the baby* Agent V Product Prep(for) Beneficiary
Benefactive Alternation (for variant) *Martha carved a toy out of a piece of wood for the baby* Agent V Product Prep(from out of) Material Prep(for) Beneficiary

Table 2.1: Benefactive alternations for the verb *carve* as recorded in VerbNet (version 1.0).

annotation of the Penn TreeBank. Although the VerbNet verb classes projected in PropBank represent powerful generalizations for the syntactic behavior of verbs, most of the time the traditional abstract semantic roles used, are too generic to capture scenarios similar to those that can be represented using a semantic frames. The difference in design leads to differences in product. For instance: The buyer of a *buy* event and the seller of a *sell* event would both be *agent*s in PropBank, while in FrameNet one is the *Buyer* and the other is the *Seller* [Ellsworth et al. 2005].

In this thesis a lexicon of unique pairs of predicators and valence patterns is extracted from the FrameNet repositories. FrameNet is chosen over PropBank as the resulting semantic representations of predicate argument structure is much more fine-grained and offers much more potential regarding reasoning tasks than representations that would result from the use of PropBank. Secondly, FrameNet captures valence patterns of non-verbs allowing semantic representations that can not be derived from PropBank.

The use of a semantically motivated resource such as FrameNet prompts the question of whether it will still be possible to map syntactic analysis of sentences, delivered by the mentioned state-of-the-art NLP techniques, to semantically motivated valence patterns. This is exactly the question that will need to be answered in the second part of this thesis.

Chapter 3

Statistical approaches to semantic role labeling

3.1 Statistical parsing models

The parsers described in both [Collins 1997] and [Hockenmaier and Steedman 2002] (left column of table 1.1 on page 10) construct syntactic derivation trees based on a set of rules that recursively *generate* well-formed English expressions. Generative models of syntax have been central in computational linguistics since they were introduced in [Chomsky 1957].

Each sentence-tree pair (S, T) in a language has an associated top-down derivation consisting of a sequence of rule applications of a grammar. These models can be extended to be statistical by defining probability distributions at points of non-determinism in the derivations, thereby assigning a probability $P = (S, T)$ to each (S, T) pair. The parser itself is an algorithm which searches for the tree, T_{best}, that maximizes $P(T \mid S)$.

A generative model uses the observation that maximizing $P(T, S)$ is equivalent to maximizing $P(T \mid S)$:

$$T_{best} = argmax_T P(T \mid S) = argmax_T \frac{P(T, S)}{P(S)} = argmax_T P(T, S)$$

This is true as $P(S)$ is constant.

The sets of rules, the grammar, used to build well-formed expressions can be of rather different nature. [Collins 1997] uses a lexicalized context free grammar, while [Hockenmaier and Steedman 2002] propose to use a specific categorial grammar. It goes beyond the scope of this thesis to fully describe both grammar formalisms and their relations. The derivation trees that result from adoption of both grammars will nevertheless be analyzed to illustrate how a semantic role labeler can benefit from them.

3.2 Statistical semantic role labeling

The approaches proposed by both [Gildea and Jurafsky 2002] and [Gildea and Hockenmaier 2003] (right column of table 1.1 on page 10) estimate, based on

some training corpus, the probability distribution of the occurrence of a role r_j given all combinations of features $F_{1...n}$. The features regard both the constituent i and the target predicator p:

$$P(r_j \mid pt_i, voice_p, path_{ip}, head_i, pred_p)$$

With:

- $r_j \in R$, with R the set of semantic roles;
- $pt_i \in PT$, with PT the set of phrase types of (NP, PP, VP, S etc.);
- $voice_i \in VOICE$, with $VOICE$ the set of voices ($\{active, passive\}$);
- $path_i \in PATH$, with $PATH$ the set of all possible 'syntactic relations' between constituent i and a predicator p;
- $head_i \in HEAD$, with $HEAD$ the set of words that may occur as the head of the constituent;
- $pred_i \in PRED$, with $PRED$ the set of predicates that can be denoted by the target verb form.

Theoretically the set of feature combinations for which, per role, a probability will need to be determined, is immense:

$$|F| = |PT| \times |VOICE| \times |PATH| \times |HEAD| \times |PRED|$$

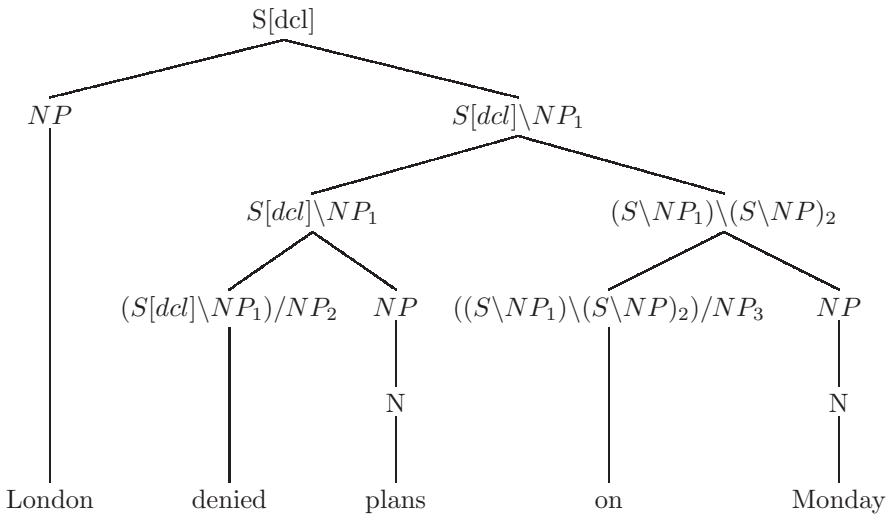
Although not all possible feature combinations will occur in natural language, the subset of combinations that do, will be very large. This is mainly due to the amount of possible predicates (—*PRED*—), the amount of words that can appear as head word of a constituent (—*HEAD*—) and the number of possible predicator-constituent relations in parse trees (—*PATH*—).

One should realize that a fine-grained set of almost predicate specific semantic roles, such as defined in FrameNet, can only be assigned on the basis of actually knowing which predicate *pred* is dealt with or at least to which frame this predicate belongs. One way to distinguish between for instance *He bought a car* (a commercial transaction) and *He bought it* (an act of deceiving), is by recognizing that a constituent with head word *car* is not likely to fill the role of *Falsehood* in the *Deceiving* frame. One might argue that the task of assigning semantic roles to sentence constituents will often be dependent on some sort of word-sense disambiguation in which the *HEAD* and *PRED* features are of great importance.

The *PATH* feature is designed to capture the syntactic relation of a constituent to the predicator. This relation is determined in a derivation tree produced by a syntactic parser. Two possible 'path' representations are defined in figure 3.2 and 3.1.

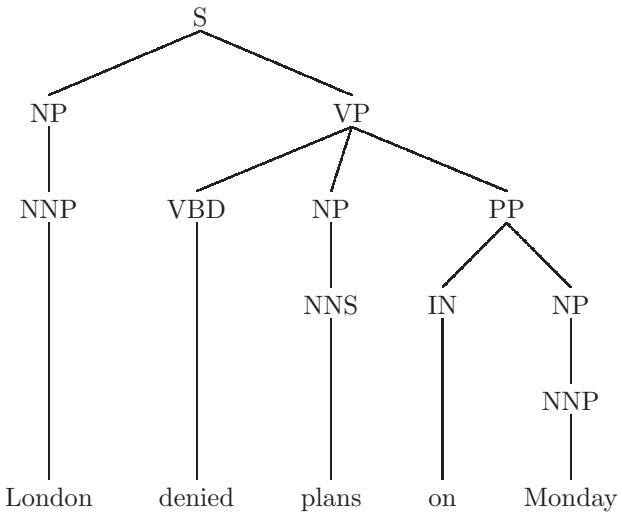
In the Penn TreeBank derivation predicate-argument structure is not explicitly determined. The relation between two words can (only) be represented by the shortest route through the derivation tree that connects them. Certain paths might be typical for a particular predicate-argument relation, but there is no way assure this relation.

When analyzing *London* ***had*** *denied plans on Monday* the following paths between *London* and *denied* will be found using the two parsers:



	w	w'		
path	denied	plans	=	$\langle (S[dcl]\backslash NP)/NP, 2, \leftarrow \rangle$
path	denied	on	=	$\langle ((S\backslash NP)\backslash (S\backslash NP))/NP, 2, \rightarrow \rangle$
path	denied	London	=	$\langle (S[dcl]\backslash NP)/NP, 1, \leftarrow \rangle$

Figure 3.1: The path feature relating the predicator (or functor) *denied* to the other constituents of the sentence as determined by a CCG derivation. *London* and *plans* are the heads of the constituents that denote the arguments of the predicate denoted by *denied*, while *on* is the head of the constituent that modifies this predicate. The relation between words w and w' is determined by the category of the predicator ($(S\backslash NP_1)/NP_2$), the index of the 'argument' filled by the constituent and an arrow to indicate whether w ($\leftarrow$) or w' ($\rightarrow$) is the predicator.



	w	w'		
path	denied	plans	=	$VBD \uparrow VP \downarrow NP$
path	denied	on	=	$VBD \uparrow VP \downarrow PP \downarrow IN$
path	denied	London	=	$VBD \uparrow VP \uparrow S \downarrow NP$

Figure 3.2: The path feature relating the predicator *denied* to the other constituents of the sentence as determined by a Penn TreeBank style derivation. Again *London* and *plans* are the heads of the constituents that denote the arguments of the predicate denoted by *denied*, while *on* is the head of the constituent that modifies this predicate. This is however no longer explicitly specified.

- (CCG) $\langle (S[dcl]\backslash NP)/NP, 1, \leftarrow \rangle$
- (Penn) $\langle VBD \uparrow VP \uparrow \mathbf{VP} \uparrow S \downarrow NP, \leftarrow \rangle$

The path feature determined from the CCG derivation is equal to that in found in figure 3.1. The addition of the auxiliary *had* has no influence on the value of the path function. The Penn TreeBank style derivation determines a path that differs from the one found in figure 3.2. This difference will also occur in sentences with long distance dependencies (*[The man] who is thought to [love] Mary ...*) or cases of coordination (*[I] loathe and [detest] opera*). This illustrates how the use of CCG will produce a much smaller *PATH* set and thus a smaller number of probability distributions needed to be estimated. Section 4 will more extensively document the differences.

In fact this very effect, the size of *PATH*, is investigated in [Gildea and Hockenmaier 2003] by an analysis of the performance of two semantic role labelers. The only difference between the two was the chosen path feature. Both the training and test sets were equal for both labelers. As expected the CCG-based approach performed better than the approach using Penn TreeBank style parses.

In general stronger generalization regarding any of the features in F will make performance less sensitive to sparsity of training data, causing higher coverage. The danger of achieving higher coverage at the cost of lower accuracy is often present. Regarding the path feature this danger is cancelled by the grammar formalism that just more accurately expresses predicate-argument structure.

3.3 Sparsity

In order to deal with the sparsity of training data when estimating probabilities $P(r_i \mid F_j)$ [Gildea and Jurafsky 2002] proposes and compares several approaches. In many cases, a particular combination of features will never be seen in the training data, and in others a combination will only occur a small number of times, providing a poor estimate of the probability. The small number of training sentences for each target word and the large number of values that the $HEAD$ feature in particular can take (any word in the language) contribute to the sparsity of the data. This means often the probability of a role can only be estimated by combining probabilities from distributions conditioned on a variety of subsets of F.

In order to combine the strengths of the various distributions, they can be merged in various ways to obtain an estimate of the full distribution

$$P(r_j \mid pt_i, voice_p, path_{ip}, head_i, pred_p)$$

The first combination method is *linear interpolation*, which simply averages the probabilities given by each of the chosen distributions. E.g.:

$$\begin{aligned} P(r \mid constituent) = {} & \lambda_1 P(r \mid pred) + \lambda_2 P(r \mid pt, pred) + \\ & \lambda_3 P(r \mid pt, path, pred) + \lambda_4 P(r \mid pt, voice) + \\ & \lambda_5 P(r \mid pt, voice, pred) + \lambda_6 P(r \mid pred) + \ldots \end{aligned}$$

Other approaches are to calculate the *geometric mean*[1] or, more sophisticated, to choose interpolation weights using the *Expectation Maximization (EM) algorithm*[2]EM alternates between performing an expectation (E) step, which computes an expectation of the likelihood by including the latent variables as if they were observed, and a maximization (M) step, which computes the maximum likelihood estimates of the parameters by maximizing the expected likelihood found on the E step. The parameters found on the M step are then used to begin another E step, and the process is repeated.).

Another combination method uses a lattice to select a subset of the available distributions to combine. In this case the less specific distributions are used only when no data is present for any more specific distribution. The selected probabilities can again be combined with both linear interpolation and a geometric mean.

[Gildea and Jurafsky 2002] investigates the influence of the mentioned merging approaches and concludes that differences in percentage of correctly predicted roles are very small (1% or less).

It needs to be mentioned that the distributions calculated are simply the empirical distributions from the training data. That is, occurrences of each role and each set of conditioning events are counted, and probabilities calculated by dividing the counts for each role by the total number of observations for each conditioning event. For example, the distribution $P(r \mid pt, pred)$ would be calculated as follows:

$$P(r \mid pt, pred) = \frac{\#(r, pt, pred)}{\#(pt, pred)}$$

In general the sparsity of training data causes probabilities to be estimated from combinations of distributions that are (too) little specific. This means that in order to be able to estimate the probability of a certain semantic role, there needs to be (too) much reliance on little information. This is a fundamental problem of stochastic approaches to many machine learning tasks.

[1]The geometric mean of a set of positive data is defined as the nth root of the product of all the members of the set, where n is the number of members.

[2](

Chapter 4

Combinatory Categorial Grammar

The previous section already mentioned the possibility of a more general PATH feature when using Combinatory Categorial Grammar as grammar formalism. This section discusses the benefits of using CCG, in stead of the context free grammar used in [Gildea and Jurafsky 2002], in more detail.

In CCG, as in other varieties of categorial grammar, syntactic behavior is encoded in categories. The structure of a category is derived from the meaning it expresses. Categories are either atomic (usually S, N, NP and PP) or functional. The latter means that in CCG elements like verbs are associated with a category that specifies the *type* and *directionality* of their arguments and the type of their result. For example, the transitive verb *loves* can be a function from an (object) NP into a predicate, in turn a function from a (subject) NP into S[1]:

$$loves \vdash (S\backslash NP)/NP$$

In CCG categories combine through a small set of combinatory rules thought to be universal. Simplest are two *application rules* which allow a functional category to consume its argument either on its right ($>$) or on its left ($<$):

$$\begin{array}{llcll} P/Q & Q & \Rightarrow & P & (>) \\ Q & P\backslash Q & \Rightarrow & P & (<) \end{array}$$

Four other rules allow functions to *compose* with other functions[2]:

$$\begin{array}{llcll} P/Q & Q/S & \Rightarrow & P/S & (> \mathbf{B}) \\ Q\backslash S & P\backslash Q & \Rightarrow & P\backslash S & (< \mathbf{B}) \\ P/Q & Q\backslash S & \Rightarrow & P\backslash S & (> \mathbf{B}_\times) \\ Q/S & P\backslash Q & \Rightarrow & P/S & (< \mathbf{B}_\times) \end{array}$$

Finally there are two further rules of *type-raising* which turn an argument into a function over functions that seek that argument:

[1] In this thesis 'result leftmost' notation is used in which a rightward-combining functor over a domain β into a range α is written α/β, while the corresponding leftward-combining functor is written $\alpha\backslash\beta$.

[2] **B** stems from a similar combinator in Haskell Curry's combinatory logic [Curry and Feys 1958]

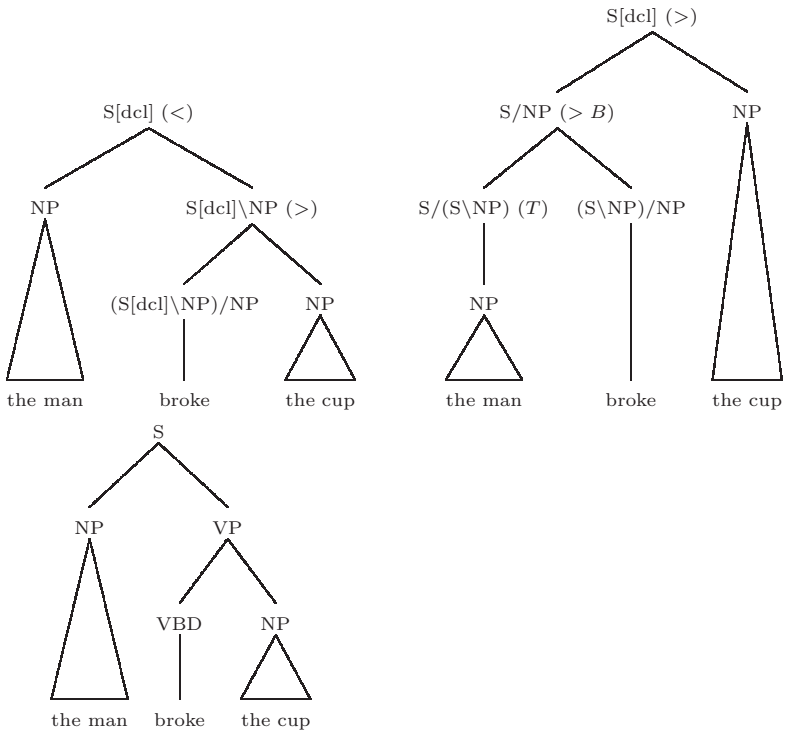


Figure 4.1: Analysis of *The man broke the cup* by the statistical CCG parser (above) and by Collins statistical tree parser (below). The applied combinatory rules are depicted between brackets next to the resulting category. See table 4.1 and the text below for more details on the features marking the categories.

$$\begin{array}{lcll} Q & \Rightarrow & P/(P\backslash Q) & (>\mathbf{T}) \\ Q & \Rightarrow & P\backslash(P/Q) & (<\mathbf{T}) \end{array}$$

Examples of syntactic derivations for *The man broke the cup.* can now be constructed as depicted in the example in figure 4.1. The corresponding Penn TreeBank style derivation is also depicted in the same figure. In both CCG derivations in the figure it can easily be seen which phrases fill which argument category of the functional category for *broke*, despite the different order of resolving.

Cases of coordination can also be treated rather elegantly. Suppose for a language there exists a category $CONJ$ and a rule for conjunction:

$$P \quad CONJ \quad P \quad \Rightarrow \quad P \qquad (<\&>)$$

Then this would allow the derivation of *I loathe and detest opera* in figure 4.2 on page 27 and even that of *I dislike and Mary likes musicals* in figure 4.3 on page 28. Both examples illustrate how predicate-argument structure can directly be derived from the derivation as the indexes on the functional categories of the predicating verbs indicate which constituents fill which arguments.

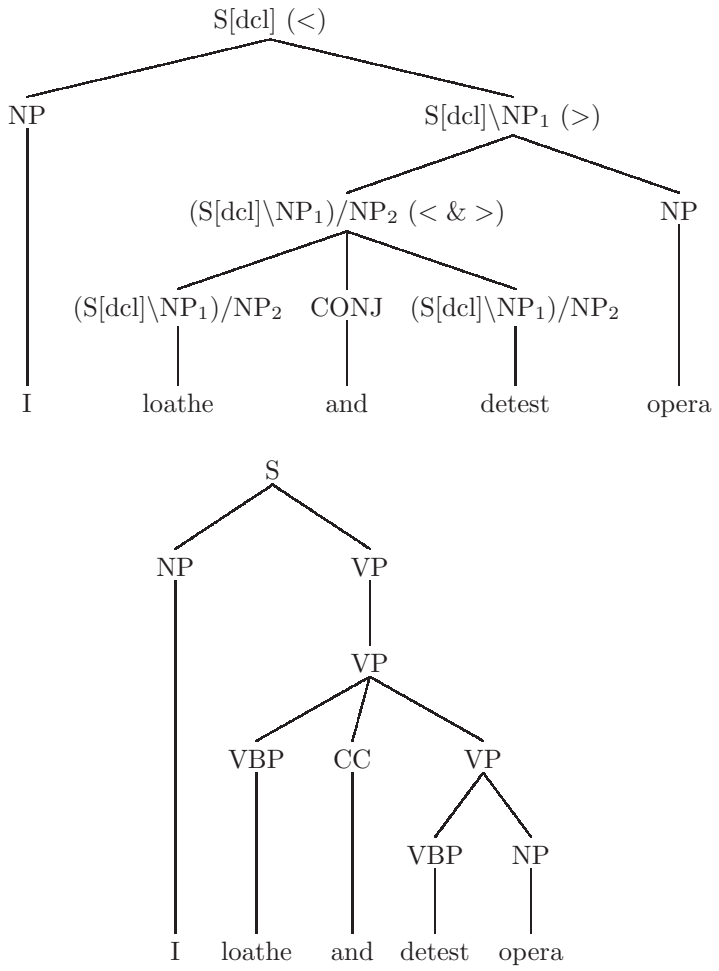


Figure 4.2: Analysis of *I loathe and detest opera* by the statistical CCG parser (top) and by Collins statistical tree parser (bottom).

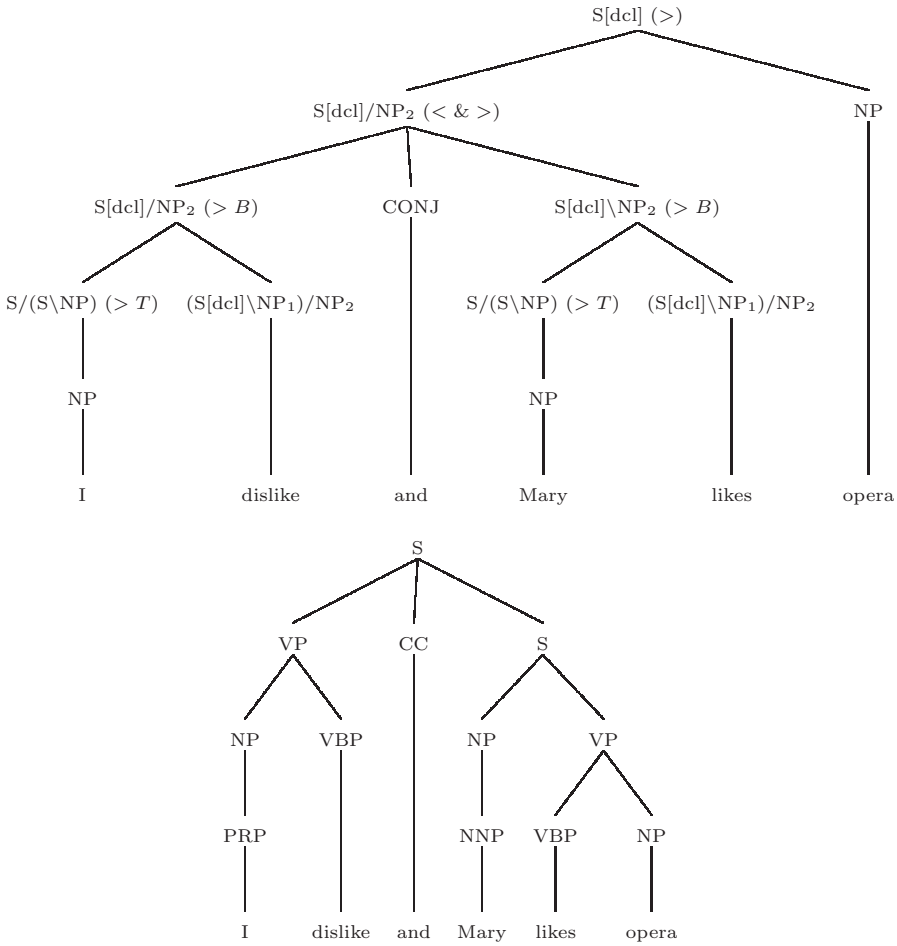


Figure 4.3: Analysis of *I dislike and Mary likes opera* by the statistical CCG parser (top) and by Collins statistical tree parser (bottom).

Finally CCG also deals with long distance dependency in a similar fashion. Figure 4 on page 30 depicts the derivations of *Who do you think broke the cup* in which *who* fills a role evoked by *broke* in the embedded sentence.

The reason for the use of the CCG variety of categorial grammar in this thesis is twofold. For one, as illustrated by the examples, CCG provides an accurate syntactic analysis of phenomena like extraction, coordination, and long distance dependencies. This allows a PATH feature that directly captures predicate argument structure. Secondly CCG has a completely transparent interface between surface syntax and underlying semantic representation. This means that providing a compositional semantics for CCG simply amounts to assigning semantic representations to the lexical entries and interpreting the combinatory rules.

The example derivations in this section correspond to the output of StatCCG, a statistical CCG parser [Hockenmaier 2003, Hockenmaier and Steedman 2002]. This parser has been trained on a preliminary version of CCGBank[3] [Hockenmaier and Steedman 2005]. The version used in this thesis comes with a pre-trained dependency model. StatCCG assumes the atomic categories S, N, NP and PP, and employs features to distinguish between declarative sentences ($S[dcl]$), wh-questions ($S[wq]$), yes-no questions ($S[q]$), embedded declaratives ($S[emb]$) and embedded questions ($S[qem]$). StatCCG further distinguishes different kinds of verb phrases ($S\backslash NP$), such as bare innitives, to-innitives, past participles in normal past tense, present participles, and past participles in passive verb phrases. This information is encoded as an *atomic feature* on the category, e.g. $S[pss]\backslash NP$ for passive verb phrase. Table 4.1 sums the atomic features that may show up in the CCG derivation trees that will be constructed as part of the role labeling algorithm presented in part two of this thesis.

[3]which differs from the one that is released by the LDC!

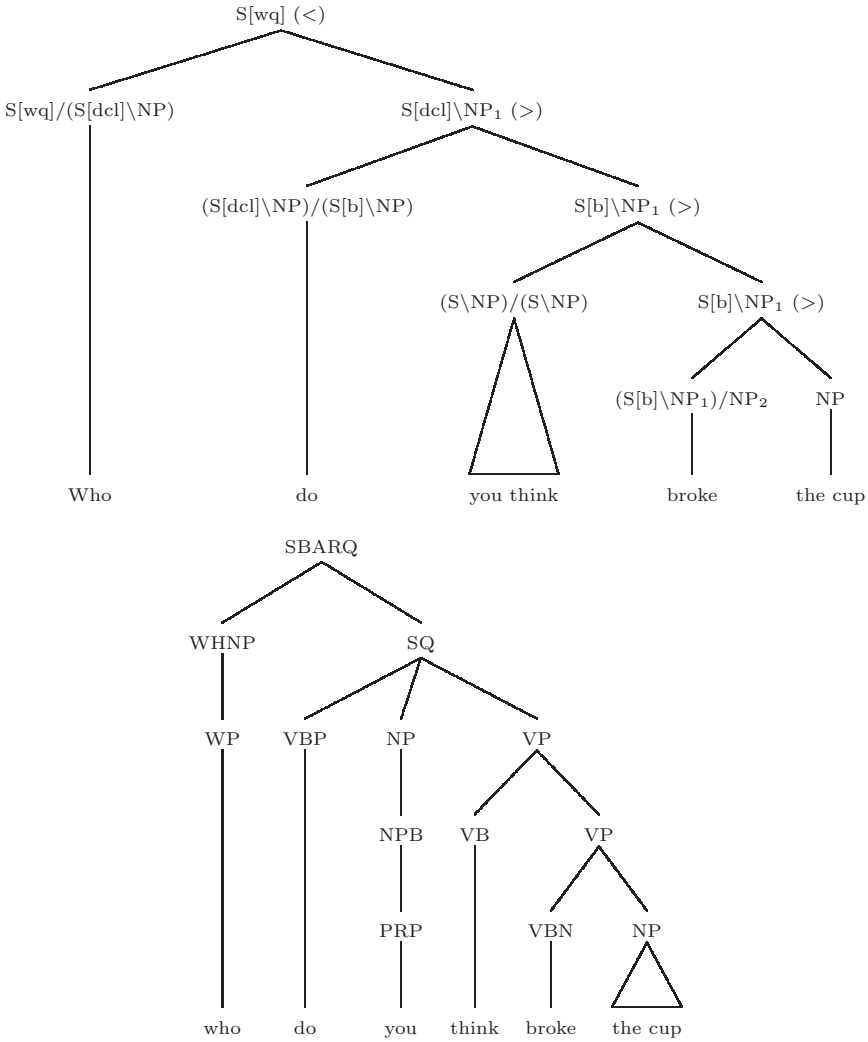


Figure 4.4: Analysis of the *Who do you think broke the cup?* by the statistical CCG parser (above) and the Collins statistical tree parser (below).

Atomic categories	
N	Noun
NP	Noun phrase
PP	Prepositional phrase
S	Sentence
Sentential features	
S[dcl]	Declarative sentences
S[wq]	Wh-questions
S[q]	Yes-no questions
S[qem]	Embedded questions
S[em]	Embedded declaratives
S[bem]	Embedded sentences in subjunctive mood
S[b]	Sentences in subjunctive mood
S[frg]	Sentence fragments
S[for]	Small clauses headed by *for*
S[intj]	Interjections
S[inv]	Elliptical inversion
Verb phrase features	
S[b]\NP	Bare infinitives, subjunctives and imperatives
S[to]\NP	To-infinitives
S[pss]\NP	Past participles in passive mode
S[pt]\NP	Past participles used in active mode
S[ng]\NP	Present participles
S[adj]\NP	Predicative adjectives
S[adj]\NP	Attributive adjective phrases

Table 4.1: CCGBank categories

Part II

Practice

Chapter 5

Task

5.1 Considered features

The main research question of this thesis was presented in chapter one: Does the use of a statistical CCG parser combined with a rule-based labeling algorithm lead to higher coverage and accuracy on a common labeling task than the use of the tree parser and stochastic labeling algorithm proposed in [Gildea and Jurafsky 2002]? In order to properly answer this question it is necessary compare the conditions of the experiments in [Gildea and Jurafsky 2002] with those in this thesis. Table 5.1 does just that with respect to the type of assigned roles and the input of both systems. Section 5.2 compares training data and test sets.

Regarding 'role level', it will only be possible to compare performance results on labeling with domain specific semantic roles. These domain specific roles are in fact equal to the desired FrameNet frame element labels. The set of abstract roles also defined in [Gildea and Jurafsky 2002] mainly serves investigation into the labeling of semantic roles evoked by predicates unseen in the training data. This (interesting) issue is not treated in this thesis.

The input of the role labeling approaches differs in more subtle ways than table 5.1 suggests. When both approaches *consider* the same feature they ob-

	[Gildea and Jurafsky 2002]	In this thesis
Role level	Abstract (custom)	not considered
	Domain specific (frame elements)	also considered
Input	Sentence	also considered
	Target	also considered
	Target voice	also considered
	Constituent phrase type	also considered
	Constituent governing category	also considered
	Constituent position	also considered
	Constituent head word	not considered

Table 5.1: Comparison of types of assigned roles and system input between the algorithm presented in this thesis and in the stochastic approach of [Gildea and Jurafsky 2002]

viously do not treat this feature in the same way. The difference between the stochastic and rule-based approach determines the consideration of the feature as a statistical classifier or a conditional respectively.

The consideration of the head word in [Gildea and Jurafsky 2002] is of minor importance to performance of the stochastic approach. This is due to the immense sparsity problem from which this feature suffers. This is the reason why the approach proposed in this thesis does not consider this feature. The feature might however, if the sparsity issue could be resolved, be the only feature that can improve semantic role labeling.

The attentive reader might have noticed the path feature (section 3.2) is missing in the table. In both the stochastic approach and approach presented in this thesis the path feature is ultimately used to determine whether a constituent is or is not a potential frame element, and not tot determine what frame element label is appropriate. For this reason the path feature is not in the table.

5.2 Test and training data

Table 5.2 provides specification of the data sets that are used to train ant test the algorithms of both approaches under investigation. [Gildea and Jurafsky 2002] 'train' by determining probability distributions of the realization of frame elements from a selection of an early version of the FrameNet repositories. In this thesis a lexicon will be extracted from a recent version of these repositories.

Both data sets are constructed by only selecting those predicators that have manually annotated sentences to illustrate the valence patterns of their frame elements. Both size and structure of the data sets are different[1].

Some remarks on the figures in table 5.2: Although the total number of annotated sentences has almost doubled between the two FrameNet releases, the average number of sentences per predicator has stayed almost equal. This means that the problem of sparsity of training data of the stochastic approach will still be apparent when using the new training data.

Secondly the increase in noun and adjective predicators has led to an increase of frames that evoke only one (or even none) frame element. This is why the total number of assigned frame elements in the training data has not not increased as spectacular as the total number of predicators.

Finally this thesis considers only those predicators that are illustrated by a minimum of 20 annotated sentences. This was necessary in order to suppress the total amount of training data, and thus the run-time of the experiments. The figures on the average number of sentences per predicator do however indicate that this number is not of much influence on the sparsity of training data that the stochastic approach would encounter on the new set.

In both approaches, the resulting corpus was divided as follows: one-tenth of the annotated sentences for each target word were reserved as a test set with a corresponding gold-standard, and another one-tenth were set aside as a tuning set for developing the systems. The rest of the sentences were used to either determine probability distributions or to construct a lexicon.

[1] It was not possible to recollect the exact set of annotated sentences used in the experiments in [Gildea and Jurafsky 2002].

	[Gildea and Jurafsky 2002]	Thesis
minimum number of annotated sentences per predicator	10	20
total number of annotated sentences in data set	49,013	97,967
total number of verb predicators	927	1,077
total number of noun predicators	339	1,041
total number of adjective predicators	175	446
total number of preposition predicators	0	4
total number of assigned frame elements	99,232	135,471
average number of sentences per predicator	34	38

Table 5.2: Specification of the FrameNet data used in [Gildea and Jurafsky 2002] and in this thesis.

5.3 Handholds for replication and system requirements

Before commencing with a detailed outline of the test set, the goldstandard, the lexicon and the actual algorithm, it is necessary to provide some hand-holds for possible future replication of the labeling task. As the processes of both preparation of auxiliary resources and the setup of the labeling algorithm have a modular character, it is possible to provide the reader with the outcome of each single step taken. This allows careful inspection of the entire chain of procedures that leads to the final results as presented in section 7.1.

The outcomes of all crucial steps are recorded on the CD-ROM that is attached to this thesis. In the outline below, there will often be referred to files on this CD-ROM. It is recommended to actually look into the files, as the description of the algorithm in this thesis will mainly be aided by pseudo-code and it might sometimes be helpful to inspect the real thing. References to files on the CD-ROM are presented in footnotes in the appropriate sections. The format of a reference is as follows:

$$CDROM - [directoryA]/[file1]$$

This represents the path of a file named *file1* in a directory named *directoryA*, which is accessible from the root of the file system on the CD-ROM.

The following file formats, all UTF-8 encoded, can be found on the CD-ROM:

- Text files, recording data that serves as input to a CCG parser and Part-Of-Speech (POS) tagger;

- XML files, recording in- and output of XSL transformations;
- XSL files, containing instructions for transformation of one XML file into another;
- Bash scripts.

Scripts and XML transformations are carefully documented through comments in their code.

The presented software has proved to run smoothly on recent Linux distributions and is expected to run on basically any Unix-like platform that supports installation of the following software[2]:

1. Java Development Kit 1.5.0 (Sun);
2. A recent GNU Bourne Again SHell;
3. TENEX C Shell, an enhanced version of Berkeley csh;
4. Stanford Log-linear Part-Of-Speech Tagger;
5. StatCCG, a statistical parser for Combinatory Categorial Grammar;
6. XMLStarlet, a Command Line XML Toolkit;
7. SED streams editor.

Besides the software, about 1GB of disk space is needed to store the FrameNet lexical database. This is only necessary when one wants to extract custom lexicons, gold standards and test sets. To obtain the FrameNet database one can request a license through the FrameNet website.

Finally it is recomended to run scripts on a system with at least 256 Mb RAM and a 1.5 ghz processor, as complicated XML transformations and loading of the probability models, take serious computational effort.

[2]Note that most software is available through pre-congured packages in the repositories of virtually any Linux distribution. The rather recent version of Java (1.5.0) is required to run the Stanford POS-tagger. Please refer to the documentation of the mentioned software packages for help on their installation.

Chapter 6

Algorithm

6.1 Short outline

In order to introduce the rule-based semantic role labeling algorithm this section provides a short walk-through of its elements. The algorithm is a cascade of procedures that process each other's output.

Given a sentence, the algorithm calculates the thematic roles evoked by a target verb in that sentence. The basic set-up of the I/O chain is depicted in table 6.1. The gray area in the bottom of the table indicates the parts of the algorithm that are developed as part of the thesis. The following parts are distinguished in the table:

- *POS-tagging*: The sentence is tagged with part-of-speech. This is done by the *Stanford log-linear part-of-speech tagger*, which tags all words and interpunction of the sentence with corresponding POS-tags. The wide-coverage tagger adopts the set of POS-tags of the Penn Treebank [Santorini 2003];

- *CCG Parsing*: The POS-tagged sentence serves as input to *StatCCG*, a statistical Combinatory Categorial Grammar parser that produces a CCG derivation of the input sentence;

- *Selection of PRFs*: A PRF is a *Potential Role Filler*, a phrase that is predicted to fulfill a semantic role evoked by the target. PRFs are selected through analysis of the CCG derivation;

- *Lexical look-up*: The target word is looked up in a lexicon extracted from the FrameNet repositories. This produces a list of entries that represent the set of semantic and syntactic combinatory possibilities – *valence patterns* – of the target in each of its senses;

- *Matching*: The lexical entry with the valence pattern that best matches the predicted PRF pattern is selected to map its semantic roles onto the PRF pattern.

In the experiment described in section 6.4, sentence-target pairs are in fact not processed one by one. A large batch of pairs is processed in one run in order

Initial input:	
Sentence	Target verb
You have to provide some money.	*provide*
↓	
POS tagging	↓
↓	
CCG Parsing	
↓	
Selection of PRFs	Lexical look-up
↘ ↙	
Matching	
↓	
output:	
target predicator in *Supply* frame	
[You $_{Supplier}$] have to ***provide*** *[some money $_{Theme}$]*	

Table 6.1: The I/O chain of the annotation algorithm.

to prevent loading of the probability models of the tagger and parser for each pair individually. This drastically cuts processing time.

Below first the construction of the test set, the corresponding gold standard and the lexicon are described. After this, the selection of PRFs, lexical look-up and matching (the gray area in table 6.1) can be analyzed.

6.2 Test-set and gold standard

In order to experiment, a test-set is needed to conduct the experiment on. In order to be able to evaluate the experiment, the results of processing of the test-set will need to be compared to a gold standard. Both the test-set and gold standard are extracted from the FrameNet repositories. As indicated in section 5.2 one-tenth of the selected sentences that illustrate the valence patterns of predicators is used to create both sets. Random selection results in a set of 9799 (out of 97967) sentence-target pairs[1] with in total 6433 unique target tokens[2].

The gold standard corresponding to the test-set of sentence-target pairs obviously also consists of 9799 records[3]. In the gold standard the phrases that fulfill frame elements evoked by the target are recorded as follows:

[1] CDROM–test-set/testSentences.txt and CDROM–test-set/targets.txt and CDROM–test-set/frameNetIDsTestSentences.txt
[2] CDROM–test-set/uniqueTargets.txt
[3] CDROM–gold standard/gold-standard.xml

```
<gold-unit>
  <text ID="1252024">We depend upon others for this kind
    of social education throughout life .</text>
  <target frame="Reliance" pos="V">depend</target>
  <fe role="Protagonist" pt="NP">We</fe>
  <fe role="Intermediary" pt="PP[upon]">upon others</fe>
  <fe role="Benefit" pt="PP[for]">for this kind of social
    education</fe>
</gold-unit>
```

The gold standard is stored in XML as it is input to future comparison with the output of the algorithm. The element fe indicates a frame element, a semantic role filler. The attribute *pt* indicates the FrameNet phrase type of the phrase fulfilling this frame element and the *role* attribute records the label of the semantic role. The nature of the FrameNet phrase type will be extensively discussed in section 6.4.2.

6.3 Lexicon

The training data used by [Gildea and Jurafsky 2002] to determine probability distributions of valence patterns of frame elements, are, in this thesis, used to create a lexicon. 80% of the selected annotated sentences is translated into lexical entries (recall 20% was used for the test and tuning sets). Each lexical entry combines a token with a valence pattern based on the FrameNet annotation of a sentence-target pair. From, for instance, the FrameNet annotation of the sentence *We depend upon others for this kind of social education throughout life*, the following verbal lexical entry can be derived:

```
<entry>
  <target frame="Reliance">depend</target>
  <fe role="Protagonist" pt="NP" location="-1"/>
  <fe role="Intermediary" pt="PP[upon]" location="1"/>
  <fe role="Benefit" pt="PP[for]" location="2"/>
</entry>
```

From the sentence *Then it is not scepticism but a withdrawal from reality* the following nominal lexical entry can be derived:

```
<entry>
  <target frame="Departing">withdrawal</target>
  <fe role="Source" pt="PP[from]" location="1"/>
</entry>
```

And from the sentence *Find out how old the place is* the following adjective lexical entry can be derived:

```
<entry>
  <target frame="Age">old</target>
  <fe role="Degree" pt="AVP" location="-1"/>
  <fe role="Entity" pt="NP" location="1"/>
</entry>
```

The lexicon that is extracted from the entire set of training data comprises of 41840 of such entries[4]. All entries have a unique combination of target (token) and valence pattern. The valence pattern consists of triples of a *role* label, a FrameNet phrase type (*pt*) and a *location*.

Location is not explicitly recorded in FrameNet, but is calculated based on the sentence that illustrates the valence pattern: A positive integer indicates a frame element realized to the right of the target, a negative integer a frame element to the left. A larger positive integer indicates a frame element further of to the right and vice versa. This abstract notion of location will later be used, together with the phrase type feature, to match lexical entries to analyses of CCG parses. The choice for these features is fundamental to the performance of the labeling algorithm.

Now it may seem strange at first sight that there is no recollection of CCG categories in the lexical entries. Although chapter 4 introduced CCG categories as accurate syntactic indicators of predicate argument structure, the less informative notion of *location* is used instead. This has a practical reason. In order to calculate the CCG categories of the target predicators all sentences in the training data would need to be parsed by StatCCG. The phrases determined to fulfill the arguments of the target predicator in the CCG derivations would then need to be matched with the phrases annotated with semantic roles in FrameNet in order to produce the desired lexical entries. This process will run into two main problems. For one, the StatCCG parser will not always produce correct syntactic analysis of a test sentence. StatCCG has a reported coverage and accuracy of 98% and 90% respectively [Hockenmaier and Steedman 2002]. This means the resulting lexicon will contain erroneous lexical entries. It would require extensive research to determine the role of erroneous entries in failing performance of the semantic role labeler. Second, and more prominent, will be the problem of mismatches between the found argument phrases in the CCG derivations and the annotated phrases in FrameNet and vice versa. Mismatches will occur due to different phrase boundaries or whenever phrases are not indicated as argument fillers in the CCG parse, but are assigned a semantic role in FrameNet. The latter will happen quite frequently as FrameNet, for example, often annotates phrases that are considered modifiers, not arguments, of the target predicator in the CCG derivations. One needs to think only of the semantic roles FrameNet indicates for *withdrawal* and *old* in the example sentences above. Both the *Source* of the *Departing* (*withdrawal*) and the *Degree* of the *Age* (*old*) will never be considered arguments of the indicated target predicators in a CCG derivation and will therefor never be assigned the corresponding semantic role.

All in all the desired recollection of CCG categories in the lexical entries would cause great difficulties in constructing (automatically) a sound lexicon. The proposed approach, using the *location* feature, does not encounter these difficulties as the lexicon is extracted directly from the FrameNet annotated sentences. This does at the same time not mean no use of CCG is made by the role labeling algorithm. The perhaps expected process of matching CCG categories of lexical entries with CCG categories found for target predicators in the test sentences is replaced by a process in which CCG derivations of the test set are used to determine phrase boundaries of phrases that might fill semantic roles. Which phrases these are, is determined by a set of rules (thus a rule-based

[4] CDROM–lexicon/lexicon.xml

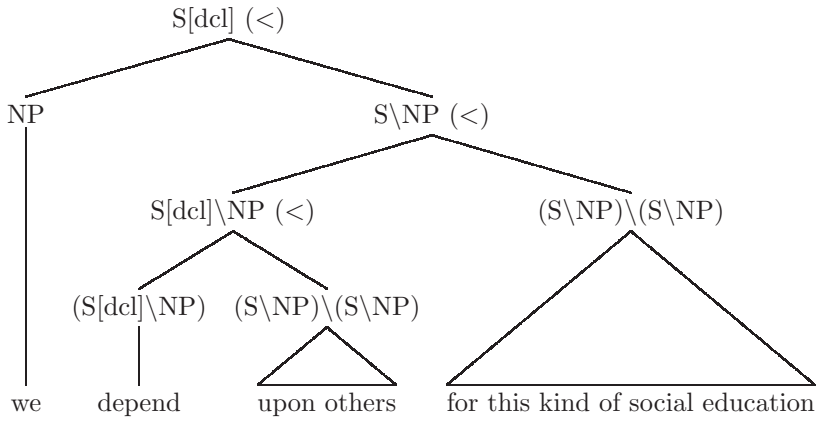


Figure 6.1: Analysis of *We depend upon others for this kind of social education* by the statistical CCG parser.

approach).

6.4 The algorithm

In section 6.1 the crucial steps of the algorithm were put forward. These were: the selection of potential frame elements (PRFs), lexical look-up and matching with lexical entries. All three steps and their relation are described in detail in this section.

In order to select PRFs, the sentences of the test-set are first POS-tagged by the Stanford log-linear part-of-speech tagger[5]. The POS-tagged sentences than serve as input to StatCCG, the statistical CCG parser. Result of parsing are 9772 CCG derivation trees for the 9799 test sentences[6] (27 or 0.3% of the sentences could not be parsed). Figure 6.1 depicts such a parse tree for the example sentence *We depend upon others for this kind of social education*.

6.4.1 Selection of Potential Role Fillers

First step of the actual algorithm is to determine which phrases in the sentence are PRFs. The phrase boundaries and the relation between phrases are determined by the CCG parser. Also the location of the target word in the derivation tree is known (and thus the abstract location of the PRFs). The algorithm in figure 6.2 determines the PRFs for each derivation tree.

Most steps of the algorithm are self-explanatory. The steps in lines `09` and `13` are treated extensively in section 6.4.2. The abstract location of a node, mentioned in line `12`, is determined in parallel with the description in section 6.3. Steps 03 to 05 express the thought that only elements that are detemined to be arguments of the functional category of the target are allowed as PRFs.

[5] CDROM–test-set/taggedTestSentences.txt

[6] CDROM–test-set/parsedTestSentences.txt and CDROM–test-set/parsedTestSentencesXML/*

```
01  select the leaf node that is marked with the target
02    for each ascendant node do
03      check whether this node is marked with
        an atomic CCG category S or NP
04        if so:  quit.
05        if not:
06          for each child node do
07            check whether this child node has a descendant
              node that is marked with the target
08              if so:  ignore this child node.
09              if not:  check whether the category of that node
                    is a category of a PRF
10                if not:  ignore the node.
11                if so:
12                  determine its astract location;
13                  determine its FrameNet phrase type;
14                  determine the phrase that is captured by all
                        descendant leaf nodes of that node;
```

Figure 6.2: Algorithm for selection of potential role fillers.

The algorithm will extract the following from the example derivation tree of the previous section:

```
<output>
  <text ID="1252024">We depend upon others for this kind
      of social education throughout life .</text>
  <target>depend</target>
  <fe pt="NP" location="-1">we</fe>
  <fe pt="PP[upon]" location="1">upon others</fe>
  <fe pt="PP[for]" location="2">for this kind of social
      education</fe>
</output>
```

6.4.2 CCG categories and FrameNet phrase types

At two points in the algorithm presented in the previous section, the CCG category of a phrase is decisive to the selection of this phrase as a PRF. First, in line 09 a CCG category has to possibly be the category of a PRF in order to be selected. This raises the question of what CCG categories might possibly denote PRFs. Secondly, in line 13 the FrameNet phrase type of the CCG category needs to be determined. This in turn demands for a possible translation of the CCG category into such a FrameNet phrase type.

As mentioned before, FrameNet has developed its own set of phrase types. These types are recorded as frame element features in the lexicon. In order to later be able to match the algorithms output with entries in the lexicon, it is necessary to translate CCG categories into FrameNet phrase types. An opposite translation, FrameNet phrase types to CCG categories, is less attractive from the perspective of the run-time of the experiment. This is due to the fact that the CCG parsing of sentences is the most time consuming step in the pro-

posed algorithm for semantic role labeling. The context of a CCG category, its composition and position in the derivation tree, are to be taken into account when translating the category into the corresponding FrameNet phrase type. In other words, CCG derivations of all sentences (80% of the FrameNet repositories) from which the lexicon was extracted would need to have been constructed and analyzed before being able to assign the 'CCG phrase type' feature to the frame elements in the lexicon. This opposed to the analysis of only the test-set sentences (10% of the selection of the FrameNet respositories).

The rules for determining whether a CCG category possibly denotes a PRF and if so, to what FrameNet phrase type it corresponds, are listed in table 6.3[7]. This set of rules is far from complete. Translations were chosen based on analysis of the CCG derivation trees of the tuning set combined with the documentation of the FrameNet phrase types in [Ruppenhofer et al. 2005] and CCG category features in [Hockenmaier and Steedman 2005]. The sections on experiment results and conclusions will elaborate on the shortcommings of the presented translation rules.

Before presenting the rules for translation, the coverage of the rules is illustrated by table 6.2. In this table all FrameNet phrase types are listed. If a translation rule for that phrase type exists, the phrase type is 'covered'. The percentage of covered phrase types gives an insight in both the extend of the translation and the effect missing translations are likely to cause on overall results in the experiment.

6.5 Lexical look-up and matching

Considering the output of selection of PRFs from the CCG derivations and the format of the lexical entries, it is not hard to imagine the possibility of matching the two:

```
<output>
  <text ID="1252024">We depend upon others for this kind
      of social education throughout life .</text>
  <target>depend</target>
  <fe pt="NP" location="-1">we</fe>
  <fe pt="PP[upon]" location="1">upon others</fe>
  <fe pt="PP[for]" location="2">for this kind of social
      education</fe>
</output>

<entry>
  <target frame="Reliance">depend</target>
  <fe role="Protagonist" pt="NP" location="-1"/>
  <fe role="Intermediary" pt="PP[upon]" location="1"/>
  <fe role="Benefit" pt="PP[for]" location="2"/>
</entry>
```

The matching algorithm is presented in figure 6.3[8]. Lexical look-up is part

[7]These rules are applied through an XSL transformation of the CCG derivation tree. CDROM–xsl/selectPRFs.xsl

[8]The matching is carried out by applying four subsequent XSL transformations. The trans-

# fe (abs)	phrase type		% covered (cummulative)
179	A	Non-maximal adjective	–
975	AJP	Standard adjective phrase	.7
1574	AVP	Adverb phrase	1.9
8936	N	Non-maximal nominal	8.5
72755	NP	Standard noun phrase	62.4
5849	Poss	Possesive noun phrase	–
34254	PP	Standard prepositional phrase	87.7
1817	PPing	Preposition with a gerund object	89.0
1	PPinterrog	Preposition with governing a wh-clause	–
1784	QUO	Quote	–
3159	Sfin	Finite clause	91.3
11	Sbrst	Finite clause	91.3
121	Sforto	For-to-marked clause	91.4
18	Sing	Gerundive clause	–
534	Sinterrog	Wh-clause	91.5
11	Sto	to-marked clause	–
111	Sub	Subordinate clause	–
61	Swhether	Whether/if clause	91.5
138	VPbrst	Bare stem verb phrase	91.5
0	VPed	Participial verb phrase	91.5
5	VPfin	Finite verb phrase	91.5
366	VPing	Gerundive verb phrase	91.6
2343	VPto	To-marked infinitive verb phrase	93.3
135002	**total**		93.3

Table 6.2: FrameNet frame elements in the selected test, tuning and training sets. The percentage of covered FrameNet phrase types, 93.3%, provides an upperbound to the algorithm performance. Only phrases with a phrase type that can be subject to some tranlation rule are possibly selected as PRFs.

CCG category	restriction(s)	FrameNet phrase type
$S[adj]\backslash NP$	–	AJP
$(S\backslash NP)\backslash(S\backslash NP)$	CCG category of a leaf node	AVP
N	phrase does not start with *it* or *there*	N
NP	–	NP
$S\backslash(S/NP)$	–	NP
$S/(S\backslash NP)$	–	NP
PP	Phrase starts with a preposition	PP[*prep*]
$N\backslash N$	Phrase starts with a preposition	PP[*prep*]
$(S\backslash NP)\backslash(S\backslash NP)$	not composed from $S[to]\backslash NP$ or $S[ng]\backslash NP$ and Phrase starts with a preposition	PP[*prep*]
PP	composed from $S[ng]\backslash NP$	PPing[*prep*]
$(S\backslash NP)\backslash(S\backslash NP)$	composed from $S[ng]\backslash NP$	PPing[*prep*]
$S[em]$	not composed from $S[em]/S[b]$, $S[bem]/S[b]$ or $S[b]/S[b]$	Sfin
$S[bem]$	not composed from $S[em]/S[b]$, $S[bem]/S[b]$ or $S[b]/S[b]$	Sfin
$S[b]$	not composed from $S[em]/S[b]$, $S[bem]/S[b]$ or $S[b]/S[b]$	Sfin
$S[b]$	–	Sbrst
$S[for]$	–	Sforto
S[wh]		Sinterrog
S[qem]	phrase does not start with *if* or *whether*	Sinterrog
S[qem]	phrase starts with *if* or *whether*	Swhether
$S[b]\backslash NP$	–	VPbrst
$S[pss]$	–	VPed
$S[pt]$	–	VPed
$S[dcl]\backslash NP$	CCG category of a non-leaf node	VPfin
$S[dcl]\backslash S[wq]$	CCG category of a non-leaf node	VPfin
$S[ng]\backslash NP$	–	VPing
$(S\backslash NP)\backslash(S\backslash NP)$	composed from $S[to]\backslash NP$	VPto
$S[to]\backslash NP$	not composed from $S[b]\backslash NP$	VPto

Table 6.3: Translation of CCG categories into FrameNet phrase types. **PRFs are those phrases that are selected by the PRF selection algorithm, have a CCG category listed in this table ánd can be translated into a FrameNet phrase type**. For information on the CCG categories and their features, the reader should refer to [Hockenmaier and Steedman 2005].

```
01  select those lexical entries that have a target that is
    equal to the target that evokes the PRFs
02  for each entry do
03    for each frame element x of that entry do
04      for each PRF y do
05        check whether the phrase type of y is equal to the
          phrase type of x
06          if not:  ignore y
07          if so:  check whether the abstract location of x
                    and y are both less than 0 AND whether
                    the abstract location of x is larger or
                    equal to that of y
08            if so:  store a record of the matching phrase
                      type, role and location of x and the
                      phrase of y
09            if not:  check whether the abstract location of
                x and y are both larger than 0 AND whether the
                abstract location of y is larger or equal to
                that of x
10              if so:  store a record of the matching phrase
                        type, role and location of x and the
                        phrase of y
11              if not:  ignore y
12  choose the entry with the largest number of records of
    role, phrase type, location and phrase, the largest
    number of matched frame elements
```

Figure 6.3: Algorithm for the matching of PRFs with lexical entries.

of this algorithm. Target, phrase type and abstract location are the features of the valence pattern that have to be matched.

Due to the complexity of the algorithm it will be hard to judge the consequences of its adoption at first sight. First consequence that can be easily checked, is the fact that matching the example entry for *depend* with the example of selected PRFs for *we depend upon others for this kind of social education*, produces a perfect match: Each frame element of the entry can be matched with a PRF.

Second consequence is the following: If two lexical entries with identical targets and valence patterns, but belonging to different frames, would be matched with a set of selected PRFs, they would have the same number of records of matched frame elements. In this case the valence patterns recorded in the lexicon would not be sufficient to distinguish between two meanings (frames) of the target[9]. If this would happen often, the chosen features for the valence pattern would need to be extended. Section 7.1 considers the extent of this problem.

The third consequence considers a multitude of situations of imperfect matches

formations are stored in CDROM–xsl/findPartTwo.xsl, CDROM–xsl/assignLE.xsl, CDROM–xsl/sortMatches.xsl and CDROM–xsl/pickBestMatch.xsl. The first file corresponds to steps $01, 02 - 11$. The other two represent step 12.

[9]In fact this situation causes a random selection of one of the best matches in step 12.

in which the number of frame elements of the lexical entry is smaller or larger than the number of predicted PRFs. The algorithm setup ensures no frame element of a lexical entry can be assigned twice. There is however another problem: The premises of steps 07 and 09 determine that if two PRFs can be matched to one frame element of a lexical entry, the first that occurs in the sentence will be chosen. Vice versa, if two frame elements of a lexical entry can be matched to one PRF, the PRF that occurs first in the sentence is chosen to match. Both situations are illustrated below. *a* is an abstract pattern of phrase type and location of a frame element in the lexical entry or PRF.

lexical entry	PRF	lexical entry	PRF
fe a ———	fe a	fe a ———	fe a
fe a			fe a

The validity of the choices made is determined in section 7.1. The problem of course only exists if no perfect match is available.

If all goes well, the matching procedure produces exactly the output that is desired from the perspective of the task at hand. Phrases of a sentence are annotated with the semantic roles evoked by the semantic frame assigned to the target[10]. Recall the perfect match of the example lexical entry with the PRFs of the example sentence:

```
<match>
  <target frame="Reliance">depend</target>
  <text ID="1252024">We depend upon others for this kind
      of social education throughout life .</text>
  <fe role="Protagonist" pt="NP" location="-1">we</fe>
  <fe role="Intermediary" pt="PP[upon]" location="1">upon
    others</fe>
  <fe role="Benefit" pt="PP[for]" location="2">for this kind of
    social education</fe>
</match>
```

6.6 Qualitative evaluation of the algorithm

Before presenting accuracy and recall results for the overall performance of the semantic role labeler, some qualitative analysis of the performance of the algorithm should be presented. It is beyond the scope, or better time-schedule, of this thesis to back-up this qualitative analysis with quantitative proof. Nevertheless the qualitative analysis should indicate the main problems of the labeling algorithm and determine whether they are of fundamental nature or not.

The algorithm considers both verbal and non-verbal target predicators. Analysis should distinguish between the two, as reasons for failing role assignment can be categorized accordingly. Second dimension of failing or problematic role assignment considers the part of the algorithm that causes the mentioned problem. Four crucial parts can be distinguished:

1. The determining of phrase boundaries and phrase relations by the CCG parser;

[10] the matching results for the test set are recorded in CDROM–results/annotatedSentences.xml

2. The 'climbing' of the derivation tree (from the node marked with the target to a node with an atomic category NP or S) that serves the PRF selection process;

3. The selection of PRFs based the existence of a fitting rule to translate a CCG category into a FrameNet phrase type;

4. The matching of PRF patterns with lexical entries.

The first item considers a part of the algorithm that was not designed as part of this thesis. Imperfect CCG derivations nevertheless occur and cause not only incorrect detection of phrase boundaries, but also subsequent incorrect relations in derivation trees. These relations affect the part of the algorithm that is indicated by the second item of the list above. If phrase boundaries are detected incorrectly, the CCG parser will most likely have determined rare categories for these phrases and determined very uncommon relations between the phrases. This means not the right phrases are tested for being a PRF in the third mentioned part of the algorithm. In this third part the set of rules for determining whether a phrase is a PRF is far from complete. Open question is whether this list can ever be perfect, but the imperfection causes PRFs to be missed or selected mistakenly. The matching of lexical entries with the found PRF patterns only (partly) fails if no perfect match exists. Alas this is all too often the case: Either the token denoting the target predicator does not exist in the lexicon or the PRF pattern is not (exactly) found in any of the lexical entries for a certain token.

Incorrect assignment of semantic roles evoked by a verbal target predicator is mainly caused by incorrect detection of phrase boundaries on the one hand and incorrect or no matching with a lexical entry on the other. Consider the selected PRFs for the sentence *We depend upon others for this kind of social education throughout life*:

```
<output>
  <text ID="1252024">We depend upon others for this kind
      of social education throughout life .</text>
  <target>depend</target>
  <fe pt="NP" location="-1">we</fe>
  <fe pt="PP[upon]" location="1">upon others</fe>
  <fe pt="PP[for]" location="2">for this kind of social
      education</fe>
</output>
```

The CCG derivation of this sentence indicates *throughout life* as a modifier of *depend* in stead of of *for this kind of social education*. FrameNet on the other hand labels the entire phrase *for this kind of social education throughout life* with the *Benefit* role. Whether FrameNet or CCG correctly analyzes the sentence is up to the reader to decide. This kind of analysis mismatch nevertheless causes many labeling errors with verbal target predicators.

The second problem of verbal target predicators considers the matching with lexical entries. As verbal targets have many different tokens that denote the same predicate, it is quite likely that an encountered token does not exist in the lexicon. If the token of the target predicator does exist not in the lexicon

no matching with a lexical entry is possible and the entire set of PRFs is not assigned any semantic role label.

The failing detection of roles evoked by non-verbal target predicators (nouns, adjectives and prepositions) is mainly caused by the first and second mentioned parts of the algorithm, although differently from verbal predicators. In CCG derivations nouns do not have a functional category. This means CCG does not treat them as predicators in the sense that FrameNet does. Consider the semantic roles evoked by the noun *argument* from the *Evidence* frame in:

Another ***argument*** *[for these diets$_{proposition}$] was [their possible role in favoring bowel rest$_{support}$]*

for these diets is selected as a PRF as it is a modifier of the noun *argument* that combines with this noun before it is combined with *another* to form the NP *Another argument for these diets*. At the moment the 'climbing' part of the algorithm reaches a CCG category NP it stops considering phrases that are related to the target predicator through branches higher up in the derivation tree. This means *their possible role in favoring bowel rest* is not detected as a PRF. Many nouns that do not only denote an instance of a certain frame but do actually evoke semantic roles suffer from this problem in sentences in which *to be* is the main verb. These nouns often have a 'verbal etymology'.

Adjectives also suffer from the same problem. *youngish*, an adjective belonging to the *Age* frame in FrameNet evokes a semantic role labeled *Entity*. This semantic role is correctly assigned to *lady* in *The youngish lady sang a song*, but *he* is not selected as a PRF, and therefore not assigned a semantic role, in *He was youngish, energetic and a good orator*.

This problem of non-verbal predicators proofs that the implementation of the climbing part of the algorithm is perhaps too simplistic. The role of the verb *to be* in sentences would need to be accommodated in order to overcome this problem.

General conclusion of this short qualitative analysis should be that the failing of the algorithm is mainly caused by a mismatch between the CCG analysis of predicate argument structure and that of FrameNet. It would be all too simple to state this mismatch is caused by the difference between semantically motivated valence patterns and a syntactic analysis of a sentence. There are many ways in which the addition of extra rules could overcome large portions of the set failed role assignments.

This qualitative analysis of course only touches the tip of the iceberg and it would be very informative to spend more effort into the research of the annotation results.

Chapter 7

Performance and conclusions

7.1 Results

The algorithm presented in the previous sections represents an extensive exercise in rule-based semantic role labeling. Motivated by the fundamental sparsity problems of the stochastic approach, this exercise outlined the basic preconditions of rule-based semantic role labeling given the set of fine-grained semantic roles and their illustrations by a resource like FrameNet.

FrameNet is not a balanced corpus of English text. The annotated sentences serve as examples of valence patterns for certain frame elements, and do not record their frequency in every day English.

Despite sparsity and frequency issues, a stochastic approach has its advantages. One might argue that the (optimal) set of rules of a rule-based system are implicitly stored in the probability distributions resulting from a simple successful training procedure. No extensive linguistic knowledge is needed to retrieve these probability distributions and experimenting with the features that can be taken into account is relatively easy and informative.

The algorithm presented in this thesis has profited from the feature experiments in [Gildea and Jurafsky 2002]. Especially the influence of the path feature, expressing the syntactic relation of a phrase to the target predicator, is investigated in many research projects regarding semantic role labeling. In fact [Gildea and Hockenmaier 2003] already prove how the performance of a stochastic role labeler benefits from the adoption of the CCG parser in stead of the tree parser of [Gildea and Jurafsky 2002]. The approach presented here can be considered to have had a head start over the stochastic approach due the adoption of the CCG grammar formalism. Table 7.1 presents the performance of both approaches on the 'common' labeling task described in section 5.2.

Recall is calculated by dividing the number of correctly labeled frame elements in the task (7538, rule-based) by the total number of frame elements that should have been labeled according to the gold standard (13460). Precision is calculated by dividing the correctly labeled frame elements by the total number of labeled frame elements (9667). The results indicate that the rule-based approach detects less frame elements, but is more successful in labeling

	[Gildea and Jurafsky 2002]	The approach of this thesis
precision	65%	78%
recall	61%	56%

Table 7.1: Precision and recall on the common labeling task.

	precision	recall
Phrase detection	81% (7807 out of 9667)	58% (7807 out of 13460)
Role detection	94% (9152 out of 9667)	68% (9152 out of 13460)
Frame detection	96% (9407 out of 9799)	94% (9212 out of 9799)

Table 7.2: Precision and recall on phrase, role and frame detection in the test-set.

the actually detected frame elements.

From the figures in table 7.2, separating detection of frames, roles and phrases, it can again be concluded, that íf a phrase, role or frame is detected, it is most likely to be correctly detected. Precision is higher than coverage. Most striking is the fact that precision and recall of phrase detection are lower than those of role detection. This indicates that roles are often detected, based on a correctly found phrase type and abstract location, despite of incorrectly predicted phrase boundaries.

All different parts of the algorithm influence the result. It would require extensive study to examine the precise role of each and every part of the cascade of procedures that annotate the sentence with semantic roles. Some basic outline of the (negative) effect of the different parts can however be provided when examining the output of the algorithm.

For one, incorrect POS tagging might cause incorrect or failing CCG parses and incorrect CCG parses might in turn cause incorrect or no PRF selection. Literature reports an accuracy of 97% for the Stanford log-linear POS tagger [Toutanova and Manning 2000] and about 90% for the CCG parser [Hockenmaier and Steedman 2002]. This means tagging and parsing will cause a loss of at most 13% $(100 - (0.97 \times 0.90 \times 100))$ of correctly assigned frame elements. In reality this figure will be much lower because it will often be possible to select some correct PRFs even from incorrect CCG parses.

Second, incorrect translation of CCG categories to FrameNet phrase types might cause incorrect or no PRF selection. After PRF selection and before matching, the precision of detected phrase boundaries is about 60%, while recall is about 83%. This is only slightly better than the precision and recall of phrase detection after matching (see table 7.2): Almost every found PRF is assigned a frame element in the matching procedure. This proves the validity of the developed matching algorithm. The small difference is mainly caused by the fact that some tokens denoting a predicate are not in the lexicon. This is due to the fact that a target token occurring in the test-set, might not have occurred in the training data and is therefor not in the lexicon. If a target token is not in the lexicon, no entry can be selected to match with the found PRFs.

Overall conclusion of the experiment should be that the (incorrect) selection of PRFs has the highest (negative) influence on the overall performance of the

role labeling algorithm.

7.2 Conclusions

The sparsity of training data caused poor estimation of probability distributions of valence patterns in [Gildea and Jurafsky 2002] and thus difficulties in semantic role assignment. This problem has not been solved by the growth of the FrameNet repository over the last few years. Table 5.2 in section 5.2 shows that the average number of illustrating annotated sentences per predicator has stayed almost equal. As thematic roles in the defined task are almost predicator specific, the stochastic approach will keep ending up with sparsity problems, a fundamental problem of stochastic approaches to NLP.

The approach presented in this thesis also runs into a rather fundamental problem, this time fundamental to rule-based approaches to NLP. It is hard to increase the extent to which a set of rules covers a problem. At some point the refinement of rules starts taking more and more effort and the benefit to the performance of the apparatus decreases with the same extend.

The results presented in the previous section do not oppose nor confirm the idea that the automatic assignment of fine-grained semantic roles is best facilitated by a rule-based approach. It has proven to be possible to achieve almost the same results with the approach presented in this thesis as was achieved by the stochastic approach of [Gildea and Jurafsky 2002].

It is fair to state that the stochastic approach has little possibility of improving its results in the future as the sparsity issue is only resolved by a much larger and more balanced corpus. FrameNet will not be that corpus as the goal of FrameNet – creating an ontology of predicators and illustrating valence patterns of semantic role fillers – will need to change in order to create the corpus desired by the stochastic approach. It is also fair to state that refinement of the rules presented for the approach of this thesis will most likely result in an considerable increase of performance of the semantic role labeler. This statement is not backed by research on refining the rules but by the assessment that a very basic set of rules, known to be incomplete, already produces results comparable to those of the stochastic approach.

Overall conclusion of this thesis should be that a resource like FrameNet, only illustrating syntactic realization of a semantically motivated set of semantic roles, is more useful to a rule-based approach to automated semantic role labeling than it is to a stochastic approach.

7.3 Future work

This thesis provided theoretical background to modern approaches to automated semantic role labeling. It further provided an exercise in putting this theory into practice. The algorithm for rule-based semantic role labeling was developed rather ad hoc. For this reason it is not recommended to try to improve the results of this particulare algorithm. It is recommended (and needed) to further study the results of the algorithm as qualitative and quantitative analysis of the results of all different steps of the algorithm will provide incentives for future approaches to rule-based semantic role labeling.

This thesis and the 'opposing' research in [Gildea and Jurafsky 2002] call for future research on a topic not touched by this thesis: semantic role assignment to roles evoked by unknown predicators. As semantic roles, or frame elements, in FrameNet are almost predicator specific, experiments on this issue should be carried out using a small set of broad semantic roles. These experiments should determine the use of rule-based and stochastic approaches to semantic role labeling in this context.

Bibliography

Collin Baker and Josef Ruppenhofer. Framenets frames vs. levins verb classes, 2002. International Computer Science Institute and University of California, Berkeley.

Patrick Blackburn. Computational semantics. *Theoria*, 18(1):27–45, 2005.

Johan Bos. Towards wide-coverage semantic interpretation. In *Proceedings of Sixth International Workshop on Computational Semantics*, 2005.

Noam Chomsky. *Syntactic Structures*. Mouton, The Hague, 1957.

Michael Collins. Three generative, lexicalised models for statistical parsing. In *In Proceedings of the 35th Annual Meeting of the ACL*, pages 16–23, Madrid, Spain, 1997. ACL02.

Haskell Curry and Robert Feys. *Combinatory Logic I*. North Holland, 1958.

Michael Ellsworth, Katrin Erk, Paul Kingsbury, and Sebastian Pad. Propbank, salsa, and framenet: How design determines product, 2005.

Charles Fillmore and Collin Baker. Framenet: Frame semantics meets the corpus, January 2000. In Poster presentation, 74th Annual Meeting of the Linguistic Society of America.

Daniel Gildea and Julia Hockenmaier. Identifying semantic roles using combinatory categorial grammar, 2003.

Daniel Gildea and Daniel Jurafsky. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288, September 2002.

Daniel Gildea and Martha Palmer. The necessity of parsing for predicate argument recognition, 2002.

Ana-Maria Giuglea and Alessandro Moschitti. Knowledge discovering using framenet, verbnet and propbank, 2004.

Jerry Hobbs, Douglas Appelt, John Bear, David Israel, Megumi Kameyama, Mark Stickel, and Mabry Tyson. Fastus: A cascaded finite-state transducer for extracting information from natural-language text. *Finite-State Language Processing*, pages 383–406, 1997.

Julia Hockenmaier. *Data and Models for Statistical Parsing with Combinatory Categorial Grammar*. PhD thesis, Institute for Communicating and Collaborative Systems, School of Informatics, University of Edinburgh, 2003.

Julia Hockenmaier and Mark Steedman. Generative models for statistical parsing with combinatory categorial grammar. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, 2002. Association for Computational Linguistics.

Julia Hockenmaier and Mark Steedman. *CCGbank: Users Manual*. University of Pennsylvania, Philadelphia, May 2005.

P. Kingsbury, M. Palmer, and M. Marcus. Adding semantic annotation to the penn treebank. In *In Proceedings of the Human Language Technology Conference*, San Diego, California, 2002.

K. Kipper, H. Dang, W. Schuler, and M. Palmer. Building a class-based verb lexicon using tags. In *In Proceedings of Fifth TAG+ Workshop*, 2000.

Beth Levin. *English Verb Classes and Alternations: a preliminary investigation*. University of Chicago Press, Chicago and London, 1993.

Mitchel Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19, 1993.

Paola Monachesi and Ineke Schuurman. The contours of a semantic annotation scheme for dutch, 2006. Dutch Language Corpus Initiative (D-Coi).

Thomas Payne. *Describing morphosyntax: A guide for field linguists*. Cambridge University Press, 1997.

Soumya Ray and Mark Craven. Representing sentence structure in hidden markov models for information extraction. In *Seventeenth International Joint Conference on Artifcial Intelligence*, Seattle, Washington, 2001. IJCAI-01.

Josef Ruppenhofer, Michael Ellsworth, Miriam R.L. Petruck, and Christopher R. Johnson. *FrameNet: Theory and Practice*. International Computer Science Institute, Berkeley, California, 2005.

Beatrice Santorini. *Part-of-Speech tagging Guidelines for the Penn Treebank Project*, 3rd revision, 2nd printing edition, June 2003.

Mark Steedman. *The syntactic process*. MIT Press, Cambridge (MA), USA, 2000. ISBN 0-262-19420-1.

Erik Tjong Kim Sang and Sabine Buchholz. Introduction to the conll-2000 shared task: Chunking. In *In Proceedings of CoNLL-2000 and LLL-2000*, Lisbon, Portugal, 2000.

Kristina Toutanova and Christopher Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *In Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, Hong Kong, 2000. EMNLP/VLC-2000.

Nianwen Xue and Martha Palmer. Automatic semantic role labeling for chinese verbs. In *In Proceedings of the 19th International Joint Conference on Artificial Intelligence*, Edinburgh, Scotland, 2005. IJCAI-05.