

# A Logical Approach to Sentiment Analysis

Niklas Christoffer Petersen



Kongens Lyngby 2012  
IMM-MSc-2012-????

Technical University of Denmark  
Informatics and Mathematical Modelling  
Building 321, DK-2800 Kongens Lyngby, Denmark  
Phone +45 45253351, Fax +45 45882673  
[reception@imm.dtu.dk](mailto:reception@imm.dtu.dk)  
[www.imm.dtu.dk](http://www.imm.dtu.dk)

© 2012 Niklas Christoffer Petersen  
IMM-MSc-2012-????

# Summary (English)

---

The goal of the thesis is to ...



# Summary (Danish)

---

Målet for denne afhandling er at ...



# Preface

---

This thesis was prepared at the department of Informatics and Mathematical Modelling at the Technical University of Denmark in partial fulfilment of the requirements for acquiring an MSc in Computer Science and Engineering.

The thesis deals with ...

The thesis consists of ...

Furthermore three project-specific learning objectives for the project concerned should be presented. The following is simply suggestions for those:

- Understand and extend modern techniques for processing of natural language texts using formal logical systems.
- Demonstrate methods for formal reasoning with respect to natural language understanding.
- Present a *proof of concept* system, that is a fully functional implementation of essential theoretical presented methods.

Niklas Christoffer Petersen



# Acknowledgements

---

I would like to thank my....



# Contents

---

<b>Summary (English)</b>	<b>i</b>
<b>Summary (Danish)</b>	<b>iii</b>
<b>Preface</b>	<b>v</b>
<b>Acknowledgements</b>	<b>vii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Classical data collection . . . . .	2
1.2 Natural language data collection . . . . .	3
1.3 Related work . . . . .	6
1.4 Using real data sets . . . . .	8
<b>2 Sentiment analysis</b>	<b>9</b>
2.1 Tokenization . . . . .	10
2.2 Lexical-syntactic analysis . . . . .	11
2.3 Mildly context-sensitive grammars . . . . .	12
2.4 Semantic analysis . . . . .	13
2.5 Data acquisition . . . . .	15
2.6 Test dataset . . . . .	15
<b>3 Combinatory categorial grammar</b>	<b>17</b>
3.1 Combinatory rules . . . . .	19
3.2 Coordination . . . . .	23
3.3 Features and agreement . . . . .	24
3.4 Extending the semantics . . . . .	25

---

<b>4</b>	<b>Lexicon acquisition and annotation</b>	<b>31</b>
4.1	Maximum entropy tagging . . . . .	32
4.2	Annotating the lexicon . . . . .	33
4.3	Calculating sentiment values . . . . .	34
<b>5</b>	<b>Implementation</b>	<b>37</b>
5.1	WordNet interface . . . . .	38
<b>6</b>	<b>Test results</b>	<b>39</b>
<b>7</b>	<b>Discussion</b>	<b>41</b>
7.1	Future work . . . . .	41
<b>8</b>	<b>Conclusion</b>	<b>43</b>
<b>A</b>	<b>A naive attempt for lexicon acquisition</b>	<b>45</b>
A.1	Tokenizer and tagger . . . . .	46
A.1.1	Shift-reduce parser . . . . .	46
A.2	Find a good title . . . . .	46
	<b>Bibliography</b>	<b>49</b>

## CHAPTER 1

# Introduction

---

In the UK, online retail sales account for over 10% of purchases, and their growth rate is markedly outstripping store-based sales [13]. Many customers express their opinions about products through online reviews. These reviews are key for marketing intelligence since they contain valuable information. Popular products may have hundreds of reviews making it hard for the customer to find the information they require, and as a result there is a need to automatically classify this data. This can benefit both customers and manufacturers of products. Customers can see what other consumers thought about the products', viewing the products strengths and weaknesses. Manufacturers can then see where their product falls short in order to improve it, and also they can compare their products to other competitive products

Cite, rewrite, *Sentiment Analysis of Customer Reviews ...*

Noisy unstructured text data is generated in informal settings such as online chat, emails, blogs, customer feedbacks and reviews. These texts have the potential to act as rich sources for raw inputs to market research and knowledge discovery. Since Internet is a crucial driving force in today's world, these texts are rich pointers to the collective opinion of the global population on almost every topic.

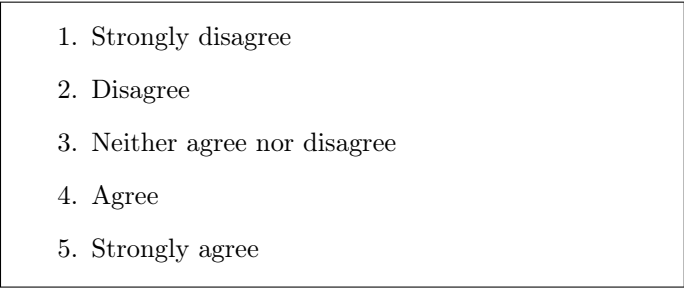
Cite, rewrite, *Opinion Mining From Noisy Text Data*

Feedback, in form of product and service reviews, can thus be highly valuable information for companies. Also political opinions are of high value for both governments and their the oppositions. The interest in such opinions is far from recent, and is a well established subfield of the *psychometrics* and has strong scientific grounds in both psychology and statistics.

Introduce *sentiment analysis* (also sometimes referred as *opinion mining*)

## 1.1 Classical data collection

One of the most used approaches to collect data for opinion analyses is through questionnaire surveys. Most of us are familiar with such surveys, where the subject is forced to answer questions with a fixed scale. For instance, given the statement “The rooms at the Swissôtel Hotel are of high quality.”, a subject must answer by selecting one of a predefined set of answers, e.g. as shown in figure 1.1

- 
1. Strongly disagree
  2. Disagree
  3. Neither agree nor disagree
  4. Agree
  5. Strongly agree

**Figure 1.1:** Likert scale.

Such scales, where the subject indicates the *level of agreement*, are known as *Likert scales*, originally presented by Likert [1932], and has been one of the favourite methods of collection data for opinion analyses cf. [?]. Other scales are also widely used, for instance the *Guttman scale* [?], where the questions are binary (yes/no) and ordered such that answering yes to a question implies the answer yes to all questions ordered

below this. Thus the answer on a Guttman scale can be captured by a single index. An example of an Guttman scale is shown in figure 1.2.

1. I like eating out
2. I like going to restaurants
3. I like going to themed restaurants
4. I like going to Chinese restaurants
5. I like going to Beijing-style Chinese restaurants

**Figure 1.2:** Guttman scale.

Given a set of answers, the result of such surveys are fairly easy to compute. At its simplest it can be an average of the answers, however mostly it is also interesting to connect the questions – for instance how does subjects’ answer to the above statement influent their answer to the statement “The food at the Swissôtel Restaurant are of high quality.”, etc.

List some shortcomings of this approach, e.g. predefined frame for feedback etc., it is often hard to get people to participate, etc.

## 1.2 Natural language data collection

In this thesis it is argued that a far more natural way for subjects to express their opinions is through their most natural communication form, i.e. their language, either in written or spoken form.

The initiative for such data collection could be *opinion seeking queries* as the one shown in (1.1). Such queries are intended to ensure succinct reviews that clearly relate to the *entity* in question (e.g. product or service) with respect to a specific

*topic of interest*. Clearly the queries could be formulated in a more friendly and *call to action* manner, as shown in (1.2)

*Holiday Inn, London: price* (1.1)

*What do you think about pricing at the Holiday Inn, London?* (1.2)

This method might not seem that different from that of the previously mentioned Likert scales, but it still allows the reviewer to answer with a much broader sentiment and lets the reviewer argue for his/hers answer as shown in the examples (1.3-1.4).

*The price is moderate for the service and the location.* (1.3)

*Overall an above average hotel based on location and price but not one for a romantic getaway!* (1.4)

More passive sources could also be considered, including posts on social networking services and microblogging services (e.g. Facebook<sup>1</sup> and Twitter<sup>2</sup>). This though introduces the need for efficient candidate filtering, as the posts in general of cause are not constrained to a specific entity or topic of interest. However it also significantly increases the data quantity, which in turn can yield a more precise analysis. Since the author of the post might never realize that the post is being used for the purpose of opinion analysis it also raises ethical issues. Larger texts, such as blog posts, could also be considered, however the contextual aspects of large, contiguous texts often makes interpretation extramly complex, thus making it a most difficult task to extract opinions on a specific entity.

It is also worth mentioning the possibility to collect answers such as (1.3, 1.4) in spoken form, which would give perhaps the most natural interaction. However for the purpose of this thesis it is proposed to solely focus on language in written form, as spoken form introduces a lot of complexity due to the speech recognition needed, for instance efficient audio sampling and analysis, speaker dependence (e.g. dialects), etc. Further more the interpretation of spoken language is also highly complex, since the emotional perception of the speaker must be considered in order to detect for instance ironic statements.

The overall goal is thus to perform a *sentiment analysis* of a set of small, succinct, review texts with respect to a subject, and yield an normalized score.

---

<sup>1</sup>Facebook, <http://www.facebook.com/>

<sup>2</sup>Twitter, <http://www.twitter.com/>



Fit

We will present the problematics that arises when trying to extract the semantic of texts from a language with a large vocabulary, and present several computational approaches for solving (at least partly) these issues.

## 1.3 Related work

In the following notable related work on sentiment analysis are briefly presented, and based on this it is argued that there are two main approaches for sentiment analysis of written texts, namely using *formal systems* and respectively using *statistical methods*.

- **Formal approaches** With this approach the language of the texts to analyse is modeled as a formal language, i.e. using a formal grammar. This allows a syntactical analysis of the texts, yielding the structures of the texts, e.g. sentences, phrases, and words. Semantic information is then extractable by augmenting and inspecting these structures.

The result of the semantic analysis is then subject to the actual sentiment analysis ...

Cites

- **Statistical approaches** With a statistical approach a probabilistic model is constructed, and trained against a training data set, configuring the parameters of the model (of which there can be an tremendous amount of). The model is then applied to the actual data set of which an analysis is desired.

can extract features from the input text. These features can then be analyzed in order to extract semantic properties.

Cites: Language models, positional language model, HHM

Both of these has been considered as the foundation of the analysis of the text reviews, however it is proposed to follow a logic approach for several reasons:

- Given that it is a formal system, it can be modeled closely and hopefully also efficiently in the *proof of concept* implementation.
- It is questionable whether the entropy of each topic in the chosen dataset actually allows feature extraction on a significant level. Given a set of text on a topic is divided into a training set and a test set, it is doubtful that the trained model would be able to capture features from the test set, since the topics only contains approximately a hundred texts.
- The undersigned has far more experience in the field of logic systems, than in the field of statistics, thus a higher level

Notice that even though it is proposed to follow a logic approach, it is still the intention to retain the proposed solutions for misspellings and minor grammatical

errors, which utilize some probabilistic properties, e.g. *match scores*, etc. However these should be seen solely as preprocessing steps needed, in case the input text cannot be analyzed directly, e.g. as a failover precaution. Thus for perfect texts, the analysis will be purely logic.

—

## 1.4 Using real data sets

For the presented solution to be truly convincing we wish to present a *proof of concept* implementation that shows at least some of the desired capabilities. However, for such product to be demonstrated properly, real data is required. Testing in on some tiny pseudo data set constructed for the sole purpose of this demonstration would not be convincing.

Move to chap. 2?

Dealing with major grammatical errors, such as wrong word order is a much harder problem, since even small changes in for instance the relative order of subject, object, verb etc. may result in an major change in interpretation. Thus it is proposed, only to focus on minor grammatical errors such as incorrect form. Such corrections could be made achievable by linking different forms of the same word.

Blog texts on the other hand suffer from ill-composed sentences, arbitrary punctuations or insertions of characters, irrational capitalization etc. Spelling correction, abbreviation and case restoration mechanisms for noisy text have been addressed in [3, 4]. We have employed variations of these techniques for our work.

Should be 3-5 pages: Motivation, Project goals, and Thesis structure

## CHAPTER 2

# Sentiment analysis

---

For the purpose of this thesis a *sentiment analysis* is defined as follows:

**DEFINITION 2.1** A sentiment analysis  $\mathcal{A}$  is a computation on a review text  $T \in \Sigma^*$  with respect to a subject  $s \in S$ , where  $\Sigma^*$  denotes the set of all permissible texts in the language. The result is an normalized score as shown in (2.1). The yielded score should reflect the *polarity* of the subject in the text, i.e. whether the overall opinion is positive (a score close to  $\omega$ ), negative (a score close to  $-\omega$ ), or neutral (a score close to 0).

$$\mathcal{A} : \Sigma^* \rightarrow S \rightarrow [-\omega; \omega] \quad (2.1)$$

It should be evident that this computation is far from trivial, and constitutes the cornerstone of this thesis. There are several steps needed, if such computation should yield any reasonable result. As mentioned in the introduction the goal is a logical approach for achieving this. The following outlines the overall steps to be completed, their associated problematics in this process, and succinctly presents different approaches to solve each step. The chosen approach for each step will be presented in much more details in later chapters. Finally ...

Something about data acquisition of test-data

## 2.1 Tokenization

In order to even start processing natural language texts, it is essential to be able to identify the elementary parts, i.e. *lexical units* and *punctuation marks*, that constitutes a text. Deont tokenization is essential for all subsequent steps. However even identifying the different sentences in a text can yield a difficult task. Consider for instance the text (2.2) which is taken from the Wall Street Journal (WSJ) corpus [Paul and Baker, 1992]. There are six periods in it, but only two of them indicates sentence boundaries, and delimits the text into its two sentences.

*Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29. Mr. Vinken is chairman of Elsevier N.V., the Dutch publishing group.* (2.2)

The domain of small review texts allows some restrictions and assumptions, that at least will ease this issue. For instance it is argued that the review texts will be fairly succinct, and thus it seems like a valid assumption that they will consists of only a few sentences. Its is argued that this indeed is achievable by sufficient instructing and constraining the reviewers doing data collection, e.g. only allowing up to a certain number of characters. This allows sentences in such phrases to be proccesed independently (i.e. as two separete review texts).

Even with this assumption, the process of identifying the sentences, and the lexical units and punctuation marks within them, is not a trivial task. Webster and Kit [1992] criticizes the neglecton of this process, as most NLP studies focus purely on analysis, and assumes this process has already been performed. Such common assumptions might derive from English being a relatively easy language to tokenize. This is due to its space marks as explicit delimiters between words, as opposed to other languages, e.g. Chinese which has no delimiters at all. This might hint that tokenization is very language dependent. And even though English is considered simple to tokenize, a naive approach like segmenting by the occurrence of spaces fails for the text (2.3), which is also from the WSJ corpus, as it would yield lexical units such as “(or”, “perceived,” and “rate),”. Simply consider all groups of non-alphanumerics as punctuation marks does not work either, since this would fail for i.a. ordinal numbers, currency symbols, and abbreviations, e.g. “Nov.” and “Elsevier N.V.” in text (2.2). Both of these methods also fail to recognize “Pierre Vinken” and “Elsevier N.V.” as single proper noun units, which is arguably the most sane choice for such.

*One of the fastest growing segments of the wine market is the category of superpremiums – wines limited in production, of exceptional quality (or so perceived, at any rate), and with exceedingly high prices.* (2.3)

Padró *et al.* [2010] presents a framework of analytic tools, developed in the recent years, for various NLP tasks. Specially interesting is the morphological analyzer, which applies a cascade of specialized (i.e. language dependent) processors to solve exactly the tokenization. The most simple use pattern matching algorithms to recognize numbers, dates, quantity expressions (e.g. ratios, percentages and monetary amounts), etc. More advanced processing are needed for proper nouns, which relies on a two-level solution: first it applies a fast pattern matching, utilizing that proper nouns are mostly capitalized; and secondly a more advanced ... TODO Cite: (Named entity extraction using adaboost). These recognize proper nouns with accuracy of respectively 90% and over 92%. The analyzer also tries to identify lexical units that are composed of multiple words, e.g. proper nouns and idioms.

It is thus possible, by the use of this framework, to preprocess the raw review texts collected from users, and ensure that they will be tokenized into segments that are suitable for the lexical-syntactic analysis. Thus more details on the tokenizer as regards design will not be presented.

## 2.2 Lexical-syntactic analysis

The syntactic analysis determines the grammatical structure of the input texts with respect to the rules of the English language. It is expected that the reader is familiar with English grammar rules and syntactic categories, including phrasal categories and lexical categories (also called parts of speech), otherwise (TODO: APPENDIX?). As mentioned earlier it is essential that the chosen solution is able to cope with *real* data, collected from actual review scenarios. This implies a robust syntactic analysis accepting a large vocabulary and a wide range of sentence structures. In order to calculate the actual polarity it is essential to have semantic annotations on the lexical units. It is argued that a feasible and suitable solution is to use a grammar that is *lexicalized*, i.e. where the rules are essentially language independent, and the syntactic properties are derived from a lexicon. Thus the development of a lexicalized grammar is mainly a task of acquiring a suitable lexicon for the desired language.

Even though the task of syntactic analysis now is largely reduced to a task of lexicon acquisition, which will be addressed in chapter 3, there are still general concerns that are worth acknowledging. Hockenmaier *et al.* [2004, p. 108-110] identifies several issues in being able to efficiently handle natural language texts solely with lexicalized grammars, mainly due to the need for entries for various combinations of proper nouns, abbreviated terms, dates, numbers, etc. Instead they suggest to use pattern matching and statistical techniques as a preprocessing step, for which efficient components exist, which translate into reduced complexity for the actual syntactic analysis.

Even though it can be argued that the use of proper nouns and specific dates is fairly limited in review text, in that a context have already been established for the reviewer cf. section 1.2, it is still

However the domain of small review texts also introduce concerns that are absent from other domains, including the possibility of incorrect grammar and spelling, since the texts comes unedited from humans with varying English skills. A solution that would only work on *perfect texts* (i.e. texts of sentences with completely correct grammar and spelling) would not be adequate. In order to at least try to handle minor misspellings it is intended to use algorithms that can select alternatives from the lexicon. Reasons for this could be that word is simply unresent from the system's vocabulary (e.g. misspelled), or on a grammatical incorrect form (e.g. wrong person, gender, tense, case, etc.).

## 2.3 Mildly context-sensitive grammars

There exists formal proofs that some natural language structures requires formal power beyond *context-free grammars* (CFG), i.e. [Shieber, 1985] and [Bresnan *et al.*, 1982]. Thus the search for grammars with more expressive power has long been a major study within the field of computational linguistics. The goal is a grammar that is so restrive as possible, allowing efficient syntactic analysis, but still capable of capturing these structures. The class of *mildly context-sensitive grammars* are conjectured to be powerful enough to model natural languages while remaining efficient with respect to syntactic analysis cf. [Joshi *et al.*, 1990].

Different grammar formalisms from this class has been considered, including *Tree Adjunct Grammar* (TAG) [Joshi *et al.*, 1975], in its lexicalized form (LTAG), *Head Grammar* (HG) [Pollard, 1984] and *Combinatory Categorical Grammar* (CCG) [Steedman, 1998]. It has been shown that these are all equal in expressive power by Vijay-Shanker and Weir [1994]. The grammar formilism chosen for the purpose of this thesis is *Combinatory Categorical Grammar* (CCG), pioneered largely by Steedman [2000]. CCG adds a layer of combinatory logic onto pure Categorical Grammar, which allows an elegant and succinct formation of *higher-order* logical expressions directly from the syntactical analysis. It is this capability, which is very usefull with respect to the goal of sentiment analysis, that mainly made the choice. Chapter 3 will formally introduce the CCG in much more detail.

CCG IS LOGICAL!!! - i.e. is uses combinatory logic



## 2.4 Semantic analysis

The overall process of semantic analysis in the context of sentiment analysis is to identify the polarity of the subjects appearing in the text, and to relate these subjects to the subject of the sentiment analysis. The approach is to *annotate* the lexical units of adjectives and adverbs with suitable polarities, and then fold these onto the phrasal structures, yielded by the syntactical analysis, in order to identify the bindings of these polarities, i.e. which subjects they modify directly or indirectly.

There exists datasets that tries to bind a general polarity to each word in a lexicon, e.g. [Esuli and Sebastiani, 2006] and [Baccianella *et al.*, 2010]. While such might be fine for general sentiment analyses, or analysis where the subject is not predefined, it is argued that better results can be achieved by using a context specific annotation. For instance the adjective “huge” might be considered positive for a review describing rooms at a hotel, while negative for a review describing sizes of cell phones.

adj?

As already mentioned, the use of a lexicalized syntactic analysis allows the annotation to appear directly on the entries in the lexicon. A manual annotation of a large lexicon is evidently not a feasible approach. Furthermore the solution must also be generic enough so it can be applied to different review contexts with minimum efforts. The concept of *semantics networks*, as [Russell and Norvig, 2009, p. 454–456]

### Semantic networks

A semeantic network is ...

A semantic network, or frame network, is a network which represents semantic relations between concepts. This is often used as a form of knowledge representation. It is a directed or undirected graph consisting of vertices, which represent concepts, and edges.[1]

Adjectives

Adverbs - intensifiers (very)

### Using real data-sets

Continue...

## 2.5 Data acquisition

In order to successfully perform the proposed computation, and thus the sentiment analysis

### Tagged corpora / Part-of-speech tagging

Since English is

*As with English around the world, the English language as used in the United Kingdom and the Republic of Ireland is governed by convention rather than formal code: there is no equivalent body to the Académie française or the Real Academia Española, and the authoritative dictionaries (for example, Oxford English Dictionary, Longman Dictionary of Contemporary English, Chambers Dictionary, Collins Dictionary) record usage rather than prescribe it. In addition, vocabulary and usage change with time; words are freely borrowed from other languages and other strains of English, and neologisms are frequent.*

[http://en.wikipedia.org/wiki/British\\_English#Standardisation](http://en.wikipedia.org/wiki/British_English#Standardisation)

## 2.6 Test dataset

### The Opinosis Dataset

The suggested data set to use is the *Opinosis Dataset*, originally used by [Ganesan *et al.*, 2010]. The data set consists of texts from actual user reviews on a total of 51 different topics. The topics are ranging over different objects, from consumer electronics (e.g. GPS navigation, music players, etc.) to hotels, restaurants and cars. For most of the objects, reviews are covered by multiple topics. For instance a specific car is covered by the topics *comfort*, *interior*, *mileage*, *performance*, and *seats*.

It has been hard to find any real alternatives for the *Opinosis Dataset* for several reasons: Most collected reviews are commercial, and thus not free to use; furthermore the *Opinosis Dataset* also contains summarized texts for each of its topics, which are constructed by manual, human interpretation. The latter allow a straight approach

for comparison of any results the proposed system will yield.

*The hotel buffet had fabulous food.* (2.4)

*Very friendly servers and nice selection of food at a reasonable price.* (2.5)

*Room service was extortionate though, very very expensive,  
so we didnt bother, as food outlets a few minutes walk away.* (2.6)

The texts (2.4) to (2.6) show actual extracts from the data set for a topic on food quality on the Swissôtel Restaurant. While (2.4) is a valid declarative sentence, (2.5) is not, since it lacks a subject (i.e. the restaurant). A coarse review of the text in the dataset reveals that missing subjects are a repeating issue. This might not seem that odd, since many people would implicitly imply the subject from the topic that they are reviewing. Thus text missing subjects can in many cases still be considered as valid sentences with minimal effort. The text (2.6) is on the other hand missing a transitive verb (presumably *are*) from the subordinate clause. In cases where such severe grammatically errors occurs it is suggested to ignore the clause, and try only to analyse the main clause. Furthermore the text (2.6) use repeated adverbs (e.g. *very very*) to express intensification, however it should not be any major concern that a verb or adjective are modified multiple times by the *same* adverb, but the intended intensification will probably not be included in the semantic analysis. Thus formalizing such a grammar is mostly a task of designing such lexicon.

As evident from these examples far from all texts in the dataset are valid sentences.

## CHAPTER 3

# Combinatory categorial grammar

---

In this chapter the formalism of Combinatory Categorical Grammar (CCG) is introduced, and based on this applied to the proposed sentiment analysis introduced in the previous chapter. For the purpose of explaining and demonstrating CCG a small fragment of English is used. This allow the usage of a “handwritten” lexicon initially. In chapter 4 the issues related to acquiring, and analyzing with, a wide coverage lexicon are addressed. For the syntactic analysis a CCG lexicon is defined as follows:

**DEFINITION 3.1** A CCG lexicon,  $\mathcal{L}_{\text{CCG}}$ , is mapping from a lexical unit,  $w \in \Sigma^*$ , to a set of 2-tuples, each containing a lexical category and semantical expression that the unit can entail cf. (3.1), where  $\Gamma$  denotes the set of lexical and phrasal categories, and  $\Lambda$  denotes the set of semantical expressions.

$$\mathcal{L}_{\text{CCG}} : \Sigma^* \rightarrow \mathcal{P}(\Gamma \times \Lambda) \quad (3.1)$$

A *tagging* of a lexical unit  $w \in \Sigma^*$  is simply the selection of one of the pairs yielded by  $\mathcal{L}_{\text{CCG}}(w)$ . Thus given some ordered set of lexical units, which constitutes the text  $T \in \Sigma^*$  to analyse, there might exists many different taggings. This is simply due to the fact that a lexical unit can entail different lexical categories (e.g. “service” is both a noun and a verb), and different semantical expressions (e.g. the noun “service” can

both refer to assistance and tableware). The number of taggings can thus be large, but is always finite.

The set of lexical and phrasal categories,  $\Gamma$ , are of a somewhat advanced structure in the CCG presented, since it follows recent work by Baldridge and Kruijff [2003] to incorporate *modalities*. A category is either *primitive* or *compound*. The set of primitive categories,  $\Gamma_{\text{prim}} \subset \Gamma$ , is language dependent and, for the English desired to be covered by this thesis, it consists of  $S$  (sentence),  $NP$  (noun phrase),  $N$  (noun) and  $PP$  (prepositional phrase). Compound categories are recursively defined by the infix operators  $/_{\iota}$  (forward slash) and  $\backslash_{\iota}$  (backward slash), i.e. if  $\alpha$  and  $\beta$  are members of  $\Gamma$ , then so are  $\alpha /_{\iota} \beta$  and  $\alpha \backslash_{\iota} \beta$ . This allows the formation of all other lexical and phrasal categories needed. The operators are left associative, but to avoid confusion inner compound categories are always encapsulated in parentheses throughout this thesis.

The basic intuitive interpretation of  $\alpha /_{\iota} \beta$  and  $\alpha \backslash_{\iota} \beta$  is as a function that takes a category  $\beta$  as argument and yields a result of category  $\alpha$ . Thus the argument is always stated on the right side of the operators, and the result on the left. The operator determines the dictionality of the application, i.e. *where* the argument should appear relative to the function: the forward operator ( $/_{\iota}$ ) denotes that the argument must appear on the right of the function, whereas the backward operator ( $\backslash_{\iota}$ ) denotes that the argument must appear on the left. The subscript,  $\iota$ , denotes the *modality* of the operator, which is a member of a finite set of modalities  $\mathcal{M}$  and will be utilized to restrict acceptance in the next section.

The syntactic categories constitutes a type system for the semantic expressions, with a set of primitive types  $\{\tau_x \mid x \in \Gamma_{\text{prim}}\}$ . Thus, if a lexicon entry has category  $(N \backslash_{\iota} N) /_{\iota} (S /_{\iota} NP)$  then the associated semantic expression must honor this, and have type  $(\tau_{NP} \rightarrow \tau_S) \rightarrow \tau_N \rightarrow \tau_N$  ( $\rightarrow$  is right associative). This is a result of the *Principle of Categorical Type Transparency* [?, Montague, 1974]. For now it is sufficient to describe the set of semantic expressions,  $\Lambda$ , as the set of *simply-typed*  $\lambda$ -expressions,  $\Lambda'$ , cf. Definition 3.2. In section 3.4 this is extended to support the desired sentiment analysis.

**DEFINITION 3.2** The set of simply typed  $\lambda$ -expressions,  $\Lambda'$ , is defined recursively as the set of expressions  $e$ , where  $e$  is either a variable  $x$  from an infinite set of typed variables  $V = \{v_1 : \tau_{\alpha}, v_2 : \tau_{\beta}, \dots\}$ , a function abstraction, or a functional application.

$$\begin{array}{lll}
 x : \tau \in V & \Rightarrow & x : \tau \in \Lambda' & \text{(Variable)} \\
 x : \tau_{\alpha} \in V, E : \tau_{\beta} \in \Lambda' & \Rightarrow & \lambda x. E : \tau_{\alpha} \rightarrow \tau_{\beta} \in \Lambda' & \text{(Abstraction)} \\
 E_1 : \tau_{\alpha} \rightarrow \tau_{\beta} \in \Lambda', E_2 : \tau_{\beta} \in \Lambda' & \Rightarrow & (E_1 E_2) : \tau_{\alpha} \in \Lambda' & \text{(Application)}
 \end{array}$$

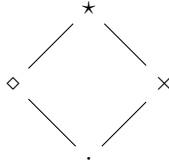
■

### 3.1 Combinatory rules

CCGs can be seen as a deductive proof system where the axioms are members of  $\Gamma \times \Lambda$ . A text  $T \in \Sigma^*$  is accepted as a sentence in the language, if there exists a deductive proof for  $S$ , for some tagging of  $T$ .

The inference rules of the proof system are known as *combinators*, since they take one or more function pairs, in the form of instances of  $\Gamma \times \Lambda$ , and produces new instances from the same set. The combinators determines the expressive power of the grammar. A deep presentation of which rules are *needed*, and thus the linguistic motivation behind this, is out of the scope of this thesis. In the following essential combinators covered by Steedman [2011, chap. 6] are succinctly described, which constitutes a *midely context-sensitive* class grammar. These are the development of the combinatory rules Steedman presented in [2000, chap. 3], however with significant changes with respect to coordinating conjunctions, due to the introduction of modalities on the infix operators.

The set of modalities used,  $\mathcal{M}$ , follows [Baldrige and Kruijff, 2003] and [Steedman, 2011], where  $\mathcal{M} = \{\star, \diamond, \times, \cdot\}$ . The set is partially ordered cf. lattice (3.2).



(3.2)

The basic concept of annotating the infix operators with  $\iota \in \mathcal{M}$ , is to restrict the application of inference rules during deduction in order ensure the soundness of the system. The  $\star$  is the most restrictive, allowing only basic rules,  $\diamond$  allows rules which perserves the word order,  $\times$  allows rules which permutate the word order, and finally  $\cdot$  allows any rule without restrictions. The partial ordering allows the most restrictive categories to also be included in the less restrictive, e.g. any rule that assumes  $\alpha/\diamond\beta$  will also be valid for  $\alpha/\star\beta$ . Since  $\cdot$  permits any rule it is convenient to simply write  $/$  and  $\backslash$  instead of respectively  $/\cdot$  and  $\backslash\cdot$ , i.e. the dot is omitted from these operators.

The simplest combinator is the *functional application*, which simply allows the instances to be used as functions and arguments, as already described. The forward and backward functional application combinator can be formulated as respectively  $(>)$  and  $(<)$ , where  $X$  and  $Y$  are variables ranging over lexical and phrasal categories, and  $f$  and  $a$  are variables ranging over semantical expressions. Since the operators are annotated with  $\star$ , the rules can apply to even the most restrictive categories. For

readability instances  $(\alpha, e)$  of  $\Gamma \times \Lambda$  is written  $\alpha : e$ . Notice that since the semantical expressions are typed, the application of  $a$  to  $f$  is sound.

$$X /_{\star} Y : f \quad Y : a \quad \Rightarrow \quad X : f a \quad (>)$$

$$Y : a \quad X \backslash_{\star} Y : f \quad \Rightarrow \quad X : f a \quad (<)$$

With only these two simple combinatory rules,  $(>)$  and  $(<)$ , the system is capable of capturing any context-free language cf. Steedman [2000, p. 34]. For the fragment of English, used to demonstrate CCG, the lexicon is considered to be finite, and it is thus possible, and also convenient, to simply write the mapping of entailment as a subset of  $\Sigma^* \times \Gamma \times \Lambda$ . Figure 3.1 shows a fragment of this demonstration lexicon. For readability, instances  $(w, \alpha, e)$  of  $\Sigma^* \times \Gamma \times \Lambda$  is written  $w \models \alpha : e$ . Notice that the semantical expressions is not yet specified, since it for now is sufficient that just the type of the expressions is correct, and this follows implicitly from the category of the entry.

<b>the</b>	$\models NP /_{\diamond} N : (\dots)$	(Determiners)
<b>an</b>	$\models NP /_{\diamond} N : (\dots)$	
<b>hotel</b>	$\models N : (\dots)$	(Nouns)
<b>service</b>	$\models N : (\dots)$	
<b>had</b>	$\models (S \backslash NP) / NP : (\dots)$	(Transative verbs)
<b>exceptional</b>	$\models N / N : (\dots)$	(Adjectives)

**Figure 3.1:** A fragment of a tiny handwritten lexicon.

The lexicon for instance shows how determiners can be modeled by the category which takes a noun on the right and yields a noun phrase. Likewise a transitive verb is modeled by a category which first takes a noun phrase on the right (the object), then a noun phrase on the left (the subject) and lastly yields a sentence. Figure 3.2 shows the deduction of  $S$  from the simple declarative sentence “the hotel had an exceptional service” (semantics are omitted).

$$\begin{array}{c}
 \frac{\frac{\frac{\text{the}}{NP /_{\diamond} N} \quad \frac{\text{hotel}}{N}}{NP} \quad \frac{\frac{\text{had}}{(S \backslash NP) / NP} \quad \frac{\frac{\frac{\text{an}}{NP /_{\diamond} N} \quad \frac{\frac{\frac{\text{exceptional}}{N / N} \quad \frac{\text{service}}{N}}{N}}{NP}}{S \backslash NP}}{S} \\
 \hline
 S
 \end{array}$$

**Figure 3.2:** Deduction of simple declarative sentence.



Besides functional application, CCG also has a set of more restrictive rules, including *functional composition*, defined by the forward and backward functional composition combinators, respectively ( $>_{\mathbf{B}}$ ) and ( $<_{\mathbf{B}}$ ), where  $Z$  likewise is variable ranging over  $\Gamma$ , and  $g$  over  $\Lambda$ .

$$\begin{aligned} X /_{\diamond} Y : f \quad Y /_{\diamond} Z : g &\Rightarrow X /_{\diamond} Z : \lambda a. f(g a) & (>_{\mathbf{B}}) \\ Y \backslash_{\diamond} Z : g \quad X \backslash_{\diamond} Y : f &\Rightarrow X \backslash_{\diamond} Z : \lambda a. f(g a) & (<_{\mathbf{B}}) \end{aligned}$$

Notice that the semantical expression yielded by ( $>_{\mathbf{B}}$ ) and ( $<_{\mathbf{B}}$ ) is equivalent to regular functional composition ( $\circ$ ) of  $f$  and  $g$ , but since  $f \circ g \notin \Lambda$  they need to be written as  $\lambda$ -expressions.

Functional composition is often used in connection with another rule, namely *type-raising*, defined by the forward and backward type-raising combinators, respectively ( $>_{\mathbf{T}}$ ) and ( $<_{\mathbf{T}}$ ), where  $T$  is a variable ranging over categories.

$$\begin{aligned} X : a &\Rightarrow T /_{\iota} (T \backslash_{\iota} X) : \lambda f. f a & (>_{\mathbf{T}}) \\ X : a &\Rightarrow T \backslash_{\iota} (T /_{\iota} X) : \lambda f. f a & (<_{\mathbf{T}}) \end{aligned}$$

Type-raising allows a, often primitive category, to ...  $\iota$  is ... (unpredictable?) and is thus often suppressed. ...

Finish

Re-write: The introduction of function composition into a categorial grammar leads to many kinds of derivational ambiguity that are vacuous in the sense that they do not correspond to semantic ambiguities.

$$\begin{array}{c} \frac{\frac{\text{the hotel}}{NP} \quad \frac{\frac{\text{provided}}{(S \backslash NP) / NP} \quad \frac{\text{a service}}{NP}}{S \backslash_{\diamond} NP} \rightarrow}{S} < \\[2ex] \frac{\frac{\frac{\text{the hotel}}{NP} \quad \frac{\text{provided}}{(S \backslash NP) / NP}}{S / (S \backslash NP)} \rightarrow_{\mathbf{T}} \quad \frac{\frac{\text{provided}}{(S \backslash NP) / NP} \quad \frac{\text{a service}}{NP}}{S /_{\diamond} NP} \rightarrow_{\mathbf{B}}}{S} < \end{array}$$

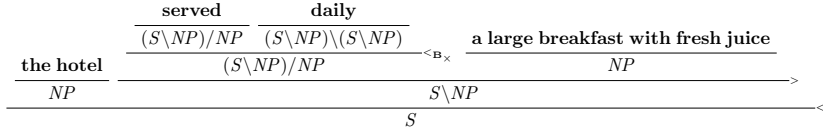
Figure 3.3: Multiple deductions.

A system with these rules demonstrates what is arguably CCG's most unique advantage, namely the ability to handle unbounded dependencies without any additional lexicon entries. For instance a transitive verb, with the *same* category as shown in figure 3.1, can participate in relative clauses as shown in Example 3.1.

$$\mathbf{that} \models (N \backslash_{\diamond} N) / (S /_{\diamond} NP) : (\dots) \quad (\text{Relative pronouns})$$

Figure 3.4: A fragment of a tiny handwritten lexicon TODO.





**Figure 3.6:** Deduction of “heavy” noun phrase shifting.

■

Steedman [2000; 2011] introduces a few additional combinators to capture even more “exotic” linguistic phenomena. Recollect that the rules are language independent, and indeed some of the additional phenomena covered by Steedman is either considered infrequent (e.g. *parasitic gaps*), or even absent (e.g. *cross-serial dependencies*), from the English language desired to cover by this sentiment analysis. It will later be shown (section ??) that the rules already presented indeed cover a substantial part of English.

## 3.2 Coordination

Coordination by appearance of a coordinating conjunction, such as *and*, *or*, *but*, punctuation and comma, etc., can be modeled simply by the intuition that such should bind two constituents of same syntactic category, but with different semantical expressions, and yield a result also of that category. Some examples of the *and* coordinating conjunction is shown in figure 3.7.

$$\begin{aligned}
 \text{and} &\models (S \backslash_{\star} S) /_{\star} S : (\dots) && (\text{Conjunctions}) \\
 \text{and} &\models (N \backslash_{\star} N) /_{\star} N : (\dots) \\
 \text{and} &\models (NP \backslash_{\star} NP) /_{\star} NP : (\dots) \\
 &\dots
 \end{aligned}$$

**Figure 3.7:** A fragment of a tiny handwritten lexicon TODO.

It now becomes evident, why the modalities are needed, since application of the crossed composition combinators without any restrictions could allow scrambled sentences to be deducted falsely, e.g. Figure 3.8.

Similar pit-falls are possible if unrestricted application of ( $>_{\mathbf{B}}$ ) and ( $<_{\mathbf{B}}$ ) was allowed,



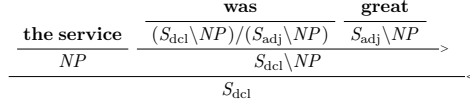


Figure 3.9: Sentence with predicative adjective.

### 3.4 Extending the semantics

The CCG presented in the previous sections has been based on established literature, but in order to apply the grammar formalism to the area of sentiment analysis the expressive power of the semantics need to be adapted to this task. Until now the semantics has not been of major concern, recall that it just was defined as simply typed  $\lambda$ -expressions cf. Definition 3.2. Furthermore the actual *semantics* of these semantic expressions has not been disclosed, other than the initial use of  $\lambda$ -expressions might hint that ordinary conventions of such presumably apply. The semantic expressions are given by Definition 3.3.

**DEFINITION 3.3** The set of semantic expressions,  $\Lambda$ , are defined as a superset of  $\Lambda'$  (see Definition 3.2). Besides variable, function abstraction and a functional application, the following structures are available: a *n*-ary *functor* ( $n \geq 0$ ) with name  $f$  from an infinite set of functor names, polarity  $j \in [-\omega; \omega]$ , and *impact argument*  $k$  ( $0 \leq k \leq n$ ); a *sequence* of  $n$  semantic expressions of the *same* type; the *change* of an expression's polarity; and the *scale* of an expression's polarity. TODO: Explain  $\psi$

$$\begin{array}{lll}
E_1, \dots, E_n \in \Lambda, 0 \leq k \leq n, j \in [-\omega; \omega] & \Rightarrow & f_j^k(E_1, \dots, E_n) \in \Lambda \quad (\text{Functor}) \\
E_1 : \tau, \dots, E_n : \tau \in \Lambda & \Rightarrow & \langle E_1, \dots, E_n \rangle : \tau \in \Lambda \quad (\text{Sequence}) \\
E : \tau \in \Lambda, j \in [-\omega; \omega] & \Rightarrow & E_{\circ j} : \tau \in \Lambda \quad (\text{Change}) \\
E : \tau \in \Lambda, j \in [-\psi; \psi] & \Rightarrow & E_{\bullet j} : \tau \in \Lambda \quad (\text{Scale})
\end{array}$$

The semantics includes normal  $\alpha$ -conversion and  $\beta$ -,  $\eta$ -reduction as shown below, where  $E_1[x \mapsto E_2]$  denotes the *safe* substitution of  $x$  with  $E_2$  in  $E_1$ , and  $FV(E)$  denotes the set of free variables in  $E$ . For details see for instance [ ].

$$\begin{array}{lll}
\lambda x.E & \Rightarrow & \lambda y.E[x \mapsto y] \quad y \notin FV(E) \quad (\alpha\text{-conversion}) \\
\lambda x.E_1 E_2 & \Rightarrow & E_1[x \mapsto E_2] \quad (\beta\text{-reduction}) \\
\lambda x.Ex & \Rightarrow & E \quad x \notin FV(E) \quad (\eta\text{-reduction})
\end{array}$$

More interesting are the rules that actually allow the binding of polarities to the phrase structures. The *change of a functor* itself is given by the rule (FC1), which apply to functors with, impact argument,  $k = 0$ . For any other value of  $k$  the functor

acts like a non-capturing enclosure that passes on any change to its  $k$ 'th argument as follows from (FC2). The *change of a sequence* of expressions is simply the change of each element in the sequence cf. (SC). Finally it is allowed to *push change* inside an abstraction as shown in (PC), simply to ensure the applicability of the  $\beta$ -reduction rule.

$$f_j^0(E_1, \dots, E_n)_{\circ j'} \Rightarrow f_{j \hat{+} j'}^0(E_1, \dots, E_n) \quad (\text{FC1})$$

$$f_j^k(E_1, \dots, E_n)_{\circ j'} \Rightarrow f_j^k(E_1, \dots, E_{k \circ j'}, \dots, E_n) \quad (\text{FC2})$$

$$\langle E_1, \dots, E_n \rangle_{\circ j'} \Rightarrow \langle E_{1 \circ j'}, \dots, E_{n \circ j'} \rangle \quad (\text{SC})$$

$$(\lambda x. E)_{\circ j'} \Rightarrow \lambda x. (E_{\circ j'}) \quad (\text{PC})$$

Completely analogue rules are provided for the scaling as shown in respectively (FS1), (FS2), (SS) and (PS). Notice that the all these *change* and *scale* rules are type preserving.

$$f_j^0(E_1, \dots, E_n)_{\bullet j'} \Rightarrow f_{j \hat{\cdot} j'}^0(E_1, \dots, E_n) \quad (\text{FS1})$$

$$f_j^k(E_1, \dots, E_n)_{\bullet j'} \Rightarrow f_j^k(E_1, \dots, E_{k \bullet j'}, \dots, E_n) \quad (\text{FS2})$$

$$\langle E_1, \dots, E_n \rangle_{\bullet j'} \Rightarrow \langle E_{1 \bullet j'}, \dots, E_{n \bullet j'} \rangle \quad (\text{SS})$$

$$(\lambda x. E)_{\bullet j'} \Rightarrow \lambda x. (E_{\bullet j'}) \quad (\text{PS})$$

It is assumed that the addition and multiplication operator, respectively  $\hat{+}$  and  $\hat{\cdot}$ , always yields a result within  $[-\omega; \omega]$ , even if the pure addition and multiplication might not be in this range cf. (3.4) and (3.5).

$$j \hat{+} j' = \begin{cases} -\omega & \text{if } j + j' < -\omega \\ \omega & \text{if } j + j' > \omega \\ j + j' & \text{otherwise} \end{cases} \quad (3.4)$$

$$j \hat{\cdot} j' = \begin{cases} -\omega & \text{if } j \cdot j' < -\omega \\ \omega & \text{if } j \cdot j' > \omega \\ j \cdot j' & \text{otherwise} \end{cases} \quad (3.5)$$

■

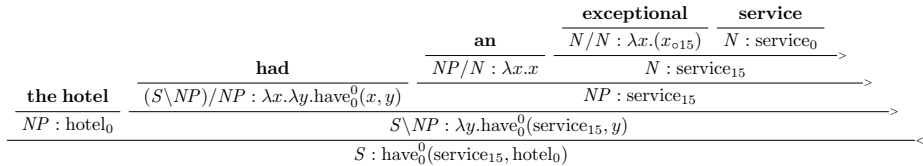
The presented definition of semantical expressions allow the specification ...

TODO: lead up to Example 3.4 and 3.4 ...

**EXAMPLE 3.3** Normally an adverb is put after the object of the verb it modifies in English, e.g. “the hotel served breakfast daily”. However if the object of the verb becomes “heavy” it may sometimes be moved to the end of the sentence, e.g. “the hotel served daily a large breakfast with fresh juice”.

*Write example text! Notice that nouns, verbs, etc. are reduced to their lemma for functor naming...*

*In such cases the adverb needs to compose with the verb, before the verb combines with its object. The crossed functional composition allows exactly such structures as shown in Figure 3.10.*

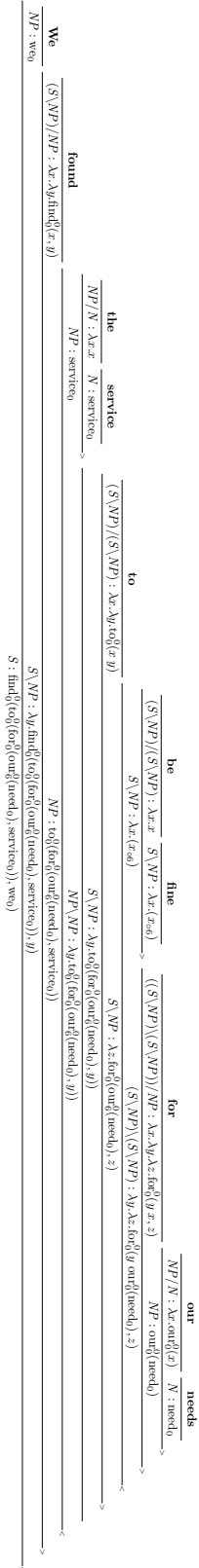


**Figure 3.10:** Deduction of “heavy” noun phrase shifting.

**EXAMPLE 3.4** Normally an adverb is put after the object of the verb it modifies in English, e.g. “the hotel served breakfast daily”. However if the object of the verb becomes “heavy” it may sometimes be moved to the end of the sentence, e.g. “the hotel served daily a large breakfast with fresh juice”.

*Write example text!*

*In such cases the adverb needs to compose with the verb, before the verb combines with its object. The crossed functional composition allows exactly such structures as shown in Figure 3.10.*



**Figure 3.1.1:** Deduction of “heavy” noun phrase shifting.



TODO: End on a up-note ...



## CHAPTER 4

# Lexicon acquisition and annotation

---

For small language fragments the lexicon can simply be “handwritten”, as demonstrated in the previous chapter. However this is obviously not an sane option when the grammar is to accept a large vocabulary and a wide range of sentence structures.

In order to use the presented model on a actual data, the acquisition of a wide covering lexicon is crucial. Initially considerable effort was made to *try* to build a CCG lexicon from a *POS-tagged corpus* (part-of-speech-tagged corpus). A POS-tagged corpus is simply a corpus where each token is tagged with a POS-tag, e.g. noun, verb, etc. There is no deep structure in such a corpus as opposed to a *treebank*. This approach turned up to have extensive issues as a result of this lack of structure, some of which are detailed in Appendix A which succinctly describes the process of the attempt.

There exists some wide covering CCG lexicons, most notable *CCGbank*, compiled by Hockenmaier and Steedman [2007] by techniques presented by [Hockenmaier, 2003]. It is essentially a translating of almost the entire Penn Treebank [Marcus *et al.*, 1993], which contains over 4.5 million tokens, and where each sentence structure has been analyzed in full and annotated. The result is a highly covering lexicon, with some entries having assigned over 100 different lexical categories. Clearly such lexicons only constitutes half of the previous defined  $\mathcal{L}_{\text{CCG}}$  map, i.e. only the lexical categories,  $\Gamma$ . The problem of obtaining a full lexicon, that also yields semantics expressions, is addressed in the next section. It is also worth mentioning that since Baldrige’s [2002]

work on modalities only slightly predates Hockenmaier’s [2003] work on CCGBank, the CCGBank does not incorporate modalities<sup>1</sup>. However more unfortunately is that CCGBank is not free to use, mainly due to license restrictions on the Penn Treebank.

What might not be as obvious is that besides obtaining a wide-covering lexicon,  $\mathcal{L}_{CCG}$ , a even harder problem is for some text  $T$  to select the *right* tagging from  $\mathcal{L}_{CCG}(w)$  for each token  $w \in T$ . Hockenmaier and Steedman [2007] calculate that the expected number of lexical categories per token is 19.2 for the CCGBank. This mean that a exhaustive search of even a short sentence (seven tokens) is expected to consider over 960 million ( $19.2^7 \approx 961.852.772$ ) possible taggings. This is clearly not a feasible approach, even if the parsing can explore all possible deductions in polynomial time of the number of possible taggings. The number of lexical categories assigned to each token needs to be reduced, however simple reductions as just assigning the most frequent category observed in some training set (for instance CCGBank) for each token is not a solution. This would fail to accept a large amount of valid sentences, simply because it is missing the correct categories.

## 4.1 Maximum entropy tagging

Clearly a solution need to base the reduction on the setting of the token, i.e. in which *context* the token appears. Clark [2002] presents a *maximum entropy model* that estimate the probability that a token is to be assigned a particular category, given the *features* of the local context, e.g. the POS-tag of the current and adjacent tokens, etc. This is used to select a subset of possible categories for a token, by selecting categories with a probability within a factor of the category with highest probability. Clark shows that the average number of lexical categories per token can be reduced to 3.8 while the parser still recognize 98.4% of unseen data. Clark and Curran [2007] presents a complete parser, which utilizes this tagging model, and a series of (log-linear) models to speed-up the actual deduction (i.e. the parsing) once the tagging model has assigned a set of categories to each token. What maybe even more interesting is that models trained on the full CCGBank, along with toolchain to use them (called the C&C tools), can be licensed freely for education or research purposes. For this reason it was chosen to use these models and tools.

Furthermore, even though the models neither incorporates modalities, since they are trained on the CCGBank, the deduction models solve many of these problems, since a more plausible deduction (i.e. a deduction seen more frequent in the CCGBank) always will suppress other less plausible deductions. Special care are taken about coordination, so neither here seems the lack of modalities to yield significant issues.

---

<sup>1</sup>There does exists another project, OpenCCG, started by Baldridge, which actually does incorporate modalities, but it has little documentation and was therefore not valued mature enough.

## 4.2 Annotating the lexicon

The output from the C&C toolchain can be printed in various formats, including Prolog, which was considered the closest to the presented model, as it, given some set of tokens,  $w_1, \dots, w_n$ , simply returns a lexicon and a deduction. A illustrative output for the tokens “the service was great” is given in Figure 4.1. In Chapter 5 more details on the actual format and the processing of it is given.

$$\begin{array}{ll}
 \alpha_1 \equiv \mathbf{the} : \text{the}_{\text{DT}} \models NP_{\text{nb}}/N & \\
 \alpha_2 \equiv \mathbf{service} : \text{service}_{\text{NN}} \models N & \\
 \alpha_3 \equiv \mathbf{was} : \text{be}_{\text{VBD}} \models (S_{\text{dcl}} \backslash NP) / (S_{\text{adj}} \backslash NP) & \frac{\frac{\alpha_1 \ \alpha_2}{NP_{\text{nb}}} > \frac{\alpha_3 \ \alpha_4}{S_{\text{dcl}} \backslash NP} >}{S_{\text{dcl}}} < \\
 \alpha_4 \equiv \mathbf{great} : \text{great}_{\text{JJ}} \models S_{\text{adj}} \backslash NP &
 \end{array}$$

(a) Lexicon
(b) Deduction

**Figure 4.1:** Illustration of output from the C&C toolchain.

Clearly, deductions in the style previously presented is trivially obtained by substituting the axioms placeholders with the lexicon entries associated. The C&C toolchain also has a build-in morphological analyzer which allow the lexicon to provide the *lemma* of the tokens. This will be convenient later. Furthermore the lexicon also provide the POS-tag<sup>2</sup> of the token which also will prove itself useful.

There are however one essential component missing from the lexicon, namely the semantic expressions.

---

<sup>2</sup>Since the C&C models are trained on CCGBank, which in turn are a translation of The Penn Treebank (PTB), the POS-tag-set used is equivalent to that of PTB.

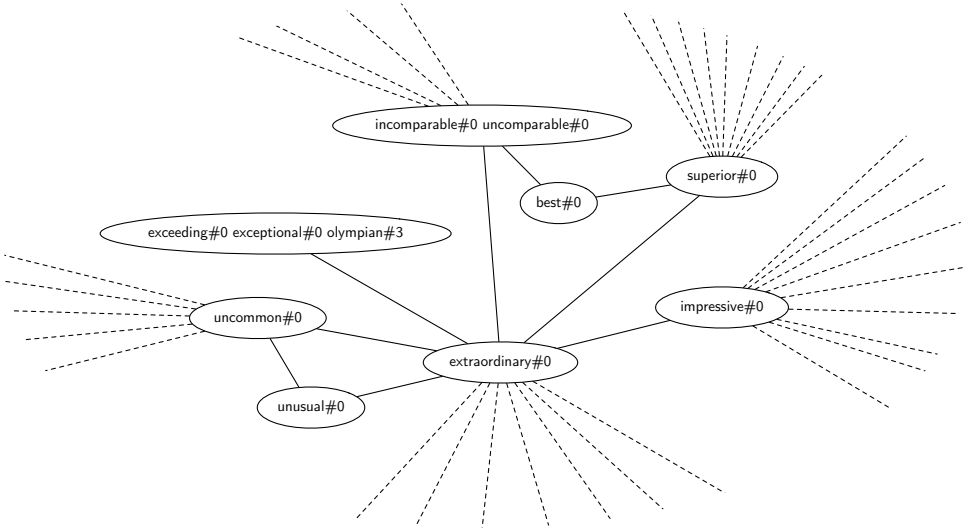
### 4.3 Calculating sentiment values

In order to reason about the polarity of the selected subject in the review text, a understanding of the domain of the review is needed. For this purpose the concept of *semantic networks* is introduced. Formally it is defined follows:

**DEFINITION 4.1** a semantic network is a quadruple  $(L, S, R, M)$  where:

- $L$  is the set of lexical units recognized by the network.
- $S$  is the set of *semantic concepts* in the network.
- $R$  is a set of binary relations on  $S$  where the relation  $r \in R$  describes links between semantic concepts, i.e.  $r \subset S \times S$ .
- $M$  is a mapping from lexical units to a set of semantic entities that the lexical unit can entail, i.e.  $M : L \rightarrow \mathcal{P}(S)$ .

Notice that  $S$  and  $R$  constitutes a set of graphs, i.e. for each relation  $r \in R$  the graph  $(S, r)$ . The graph is undirected if  $r$  is *symmetric*, and directed if  $r$  is *asymmetric*.



[Russell and Norvig, 2009, p. 453-456, 468, 471]

An illustrative example of such a graph, denoting a relation in a tiny semantic network of adjective concepts is given in figure X. The relation depicted . transitivity...

We have no way of detecting different semantic ambiguity, since we have no knowledge bade ... therefore we shall restrict our-self to





## CHAPTER 5

# Implementation

---

It was chosen to use the functional programming language *Haskell* for implementing the *proof of concept* program. In the following sections key aspects of the implementation will be presented. For the complete source code for the implementation please see appendix ??.

The reason Haskell, specifically the *Glasgow Haskell Compiler*, was chosen as programming language and platform, was the ability to ...

[van Eijck and Unger, 2010]

## 5.1 WordNet interface

The interface used to lookup data in WordNet is ... yada yada

However the interface is not complete ... the function for calculating the closure of synsets  $S$  for some relation  $R$  does not terminate if  $R$  is symmetric. Since several semantic relations are symmetric (e.g. synonyms, antonyms)

## CHAPTER 6

# Test results

---



## CHAPTER 7

# Discussion

---

Even though the structure of the sentences is inspected deeply through the syntactic analysis the semantic expression assigned to adjectives and adverbs is still atomic, and based on the key concepts the context specific positive and negative lists. Even though the lists are context specific, a concept can have ... løses dette evt. af subject-focus?

Since adjectives and adverbs are always reduced to only contribute to the polarity, they cannot be used to identify subjects. E.g. what do you think about the white iPhone vs. the black? (need better ex.!)

## 7.1 Future work

We expect such an algorithm to calculate a match score, that is a weighted average over several metrics. Given below are methods for calculating scores for some evident metrics.

- Symbolic similarity – at its most basic form we can consider a sample string (i.e. a word from an input text) against the system's vocabulary using approximate

string matching algorithms such as the *Levenshtein distance* as described by [Wagner and Fischer, 1974].

- Pronunciation similarity – it is an valid assumption that many misspellings still share a majority of the pronunciation with the intended word, i.e. they are approximately homophone. Thus comparing the phonetic properties of an sample string with possible matches can in cases correct misspellings. The *Soundex algorithm* by Robert C. Russell and Margaret K. Odell, as described by [Knuth, 1998, p. 391–92], is a simple, yet power full approach for this purpose.

---

“lift” restrictions about the texts, e.g. no of sentences ... context-sensitive, e.g. resolution of relative pronoun ... The room was luxurious, it had ...

## CHAPTER 8

# Conclusion

---





## APPENDIX A

# A naive attempt for lexicon acquisition

---

This appendix describes the efforts that was initially made in order to acquire a CCG lexicon by *transforming* a tagged corpus, namely the *Brown Corpus*.

The Brown Corpus was compiled by Francis and Kucera [1979] by collecting written works printed in United States during the year 1961. The corpus consists of just over one million words taken from 500 American English sample texts, with the intension of covering a highly representative variety of writing styles and sentence structures.

Notable drawbacks of the Brown Corpus include its age, i.e. there are evidently review topics where essential and recurring words used in present day writing was not coined yet or rarely used back 50 years ago. For instance does the Brown Corpus not recognize the words *internet*, *hotspot*

sentences will containing words has found it's way into comon that

Other corpora has been considered

## A.1 Tokenizer and tagger

The tokenizer has a very simple task, namely to convert an input string to a list of tokens (lower case words) that represent the symbols of the language. An example of the transformation is shown in (A.1).

“Put the pyramid onto the table.”  $\rightarrow$  [put, the, pyramid, onto, the, table] (A.1)

### A.1.1 Shift-reduce parser

## A.2 Find a good title

The initial attempt is simply to construct a parser that

There are three basic ways to build a shift-reduce parser. Full LR(1) (the ‘L’ is the direction in which the input is scanned, the ‘R’ is the way in which the parse is built, and the ‘1’ is the number of tokens of lookahead) generates a parser with many states, and is therefore large and slow. SLR(1) (simple LR(1)) is a cut-down version of LR(1) which generates parsers with roughly one-tenth as many states, but lacks the power to parse many grammars (it finds conflicts in grammars which have none under LR(1)).

LALR(1) (look-ahead LR(1)), the method used by Happy and yacc, is tradeoff between the two. An LALR(1) parser has the same number of states as an SLR(1) parser, but it uses a more complex method to calculate the lookahead tokens that are valid at each point, and resolves many of the conflicts that SLR(1) finds. However, there may still be conflicts in an LALR(1) parser that wouldn’t be there with full LR(1).

The state  $S_\tau$  ...

Formally a rule  $\mathcal{R}_\tau$ , for the state type  $\tau$ , is a transformation from a state  $s \in \mathcal{S}_\tau$  onto a new set of states  $\mathcal{S}'_\tau \subset \mathcal{S}_\tau$  cf. A.2.

$$\mathcal{R}_\tau : \mathcal{S}_\tau \rightarrow \mathcal{P}(\mathcal{S}_\tau) \quad (\text{A.2})$$

The state type for analysing CCGs is a 2-tuple, where  $P$  is a totally ordered set of ...,

$$\mathcal{S}_{\text{CCG}} : \mathcal{P}(T) \times \mathcal{P}(\mathcal{P}(T))$$

$$\mathcal{R}_{\text{CCG}}^{\text{shift}} \tag{A.3}$$

If all rules in the set is monotone, then the parsing will terminate



# Bibliography

---

- [Baccianella *et al.*, 2010] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*. European Language Resources Association (ELRA), 2010.
- [Baldrige and Kruijff, 2003] Jason Baldrige and Geert-Jan M. Kruijff. Multi-modal combinatory categorial grammar. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics - Volume 1*, EACL '03, pages 211–218, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.
- [Baldrige, 2002] Jason Baldrige. *Lexically Specified Derivational Control in Combinatory Categorical Grammar*. PhD thesis, University of Edinburgh, 2002.
- [Bresnan *et al.*, 1982] J. Bresnan, R. M. Kaplan, S. Peters, and A. Zaenen. Cross-Serial Dependencies in Dutch. *Linguistic Inquiry*, 13(fall):613–635+, 1982.
- [Clark and Curran, 2007] Stephen Clark and James R. Curran. Wide-Coverage Efficient Statistical Parsing with CCG and Log-linear Models. *Computational Linguistics*, 33(4):493–552, 12 2007.
- [Clark, 2002] Stephen Clark. A supertagger for combinatory categorial grammar. In *Proceedings of the 6th International Workshop on Tree Adjoining Grammars and Related Frameworks (TAG+6)*, pages 19—24, Venice, Italy, 2002.
- [Esuli and Sebastiani, 2006] Andrea Esuli and Fabrizio Sebastiani. SentiWordNet: A Publicly Available Lexical Resource for Opinion Mining. In *In Proceedings of the*

- 5th Conference on Language Resources and Evaluation (LREC'06)*, pages 417–422. European Language Resources Association (ELRA), 2006.
- [Francis and Kucera, 1979] W. Nelson Francis and Henry Kucera. *Brown Corpus Manual*. Technical report, Department of Linguistics, Brown University, Providence, Rhode Island, US, 1979.
- [Ganesan *et al.*, 2010] Kavita Ganesan, Cheng Xiang Zhai, and Jiawei Han. Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING '10)*, 2010.
- [Hockenmaier and Steedman, 2007] Julia Hockenmaier and Mark Steedman. CCG-bank: A Corpus of CCG Derivations and Dependency Structures Extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396, 2007.
- [Hockenmaier *et al.*, 2004] Julia Hockenmaier, Gann Bierner, and Jason Baldridge. Extending the coverage of a ccg system. *Journal of Language and Computation*, 2:165–208, 2004.
- [Hockenmaier, 2003] Julia Hockenmaier. *Data and Models for Statistical Parsing with Combinatory Categorical Grammar*. PhD thesis, University of Edinburgh, 2003.
- [Joshi *et al.*, 1975] Aravind K. Joshi, Leon S. Levy, and Masako Takahashi. Tree adjunct grammars. *Journal of Computer and System Sciences*, 10(1):136–163, 1975.
- [Joshi *et al.*, 1990] Aravind K. Joshi, K. Vijay Shanker, and David Weir. *The Convergence Of Mildly Context-Sensitive Grammar Formalisms*, 1990.
- [Knuth, 1998] Donald E. Knuth. *The Art of Computer Programming, vol. 3: Sorting and Searching*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 2nd edition, 1998.
- [Likert, 1932] Rensis Likert. A technique for the measurement of attitudes. *Archives of Psychology*, 22(140):1–55, 1932.
- [Marcus *et al.*, 1993] Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.
- [Padró *et al.*, 2010] Lluís Padró, Samuel Reese, Eneko Agirre, and Aitor Soroa. Semantic Services in FreeLing 2.1: WordNet and UKB. In Pushpak Bhattacharyya, Christiane Fellbaum, and Piek Vossen, editors, *Principles, Construction, and Application of Multilingual Wordnets*, pages 99–105, Mumbai, India, February 2010. Global Wordnet Conference 2010, Narosa Publishing House.

- [Paul and Baker, 1992] Douglas B. Paul and Janet M. Baker. The design for the Wall Street Journal-based CSR corpus. In *Proceedings of the workshop on Speech and Natural Language*, pages 357–362, Stroudsburg, PA, USA, 1992. Association for Computational Linguistics.
- [Pollard, 1984] Carl Pollard. *Generalized Context-Free Grammars, Head Grammars and Natural Language*. PhD thesis, Stanford University, 1984.
- [Russell and Norvig, 2009] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 3rd edition, 2009.
- [Shieber, 1985] Stuart M. Shieber. Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, 8(3):333–343, 1985.
- [Steedman, 1998] Mark Steedman. *Categorial Grammar*, 1998.
- [Steedman, 2000] Mark Steedman. *The Syntactic Process*. The MIT Press, 2000.
- [Steedman, 2011] Mark Steedman. *Taking Scope: The Natural Semantics of Quantifiers*. The MIT Press, 2011.
- [van Eijck and Unger, 2010] Jan van Eijck and Christina Unger. *Computational Semantics with Functional Programming*. Cambridge University Press, New York, NY, USA, 1st edition, 2010.
- [Vijay-Shanker and Weir, 1994] K. Vijay-Shanker and David J. Weir. The equivalence of four extensions of context-free grammars. *Mathematical Systems Theory*, 27:27–511, 1994.
- [Wagner and Fischer, 1974] Robert A. Wagner and Michael J. Fischer. The string-to-string correction problem. *J. ACM*, 21(1):168–173, January 1974.
- [Webster and Kit, 1992] Jonathan J. Webster and Chunyu Kit. Tokenization as the initial phase in nlp. In *Proceedings of the 14th conference on Computational linguistics - Volume 4*, COLING '92, pages 1106–1110, Stroudsburg, PA, USA, 1992. Association for Computational Linguistics.