Logical Entity Level Sentiment Analysis

Niklas Christoffer Petersen and Jørgen Villadsen

Department of Management Engineering
Department of Applied Mathematics and Computer Science
Technical University of Denmark, 2800 Kongens Lyngby, Denmark
`{niklch,jovi}@dtu.dk`

Abstract. We present a formal logical approach using a combinatory categorial grammar for entity level sentiment analysis that utilizes machine learning techniques for efficient syntactical tagging and performs a deep structural analysis of the syntactical properties of texts in order to yield precise results. The method should be seen as an alternative to pure machine learning methods for sentiment analysis, which are argued to have high difficulties in capturing long distance dependencies, and can be dependent on significant amount of domain specific training data. The results show that the method yields high correctness, but further investment is needed in order to improve its robustness.

1 Introduction

The amount of unstructured textual data available through the Social Web has grown rapidly over the last years. The potential in such data are numerous, and has found applications in both commercial products and services, as well as the political and financial world cf. [6].

Sentiment analysis (or opinion mining) has enjoyed high research activity for some time now, sparked by work such as [11] and [14]. Traditional approaches include statistical text classifiers and keyword-based algorithms, and will usually classify sentiment on document, sentence or simply on word level. However such granularity suffers from obvious weaknesses, for instance when trying to analyze sentences with coordination of sentiments for multiple entities, e.g. (1).

The buffet was expensive, but the view is amazing. (1)

To cope with such cases the granularity of sentiment analysis has in recent work shifted towards entity level (or concept level) approaches. However this increased degree of detail introduces new challenges, especially for statistical methods, cf. [3], both due to their semantic weakness and because labeled training data are sparse at this granularity level. Statistical methods generally rely on some fixed window for feature extraction (i.e. n-grams), and can thus fail to detect long distance dependencies between an entity and opinion stated about that entity. An illustration of this is shown by the potentially unbound number of *relative clauses* allowed in English, e.g. (2), where *breakfast* is described as *best*, however one would need to use a window size of at least 9 to detect this relation, which is arguably larger then normally considered ([11] only considers up to tri-grams).

The breakfast that was served Friday morning was the best I ever had! (2)

We present a formal logical approach for entity level sentiment analysis that utilizes machine learning techniques for efficient syntactical tagging and performs a deep structural analysis of the syntactical properties of texts in order to yield precise results.

The present paper is a substantial extension of [15]. Specifically we elabrorate on the semantic annotation and the use of semantic networks for assignment of sentiment polarity. Furthermore, we extend the method with intensifiers and qualifiers, i.e. adverbs that respectively strengthen or weaken the sentiment.

After Section 2 on related work we present in Section 3 the combinatory categorial grammar and the tagging model used. Section 4 describes the adaption to sentiment analysis. The experimental results are described in Section 5 and further discussed in Section 6. Finally Section 7 concludes.

2 Related Work

Notably related work using formal approaches includes [20] where the authors present a method of extracting sentiment from dependency structures and also focus on capturing long distance dependencies. As dependency structures simply can be seen as binary relations on words, it is indeed a formal approach. However what seems rather surprising is that in the end they only classify on sentence-level, and thus in this process loose the entity of the dependency.

The most similar work on sentiment analysis found using a formal approach is the work [17]. The paper presents a method to detect sentiment of newspaper headlines, in fact partially using the same grammar formalism that later will be introduced and used in the present work, but, however, without the combinatorial logic approach. The paper focus on some specific problems arising with analyzing newspaper headlines, e.g. such as headline texts often do not constitute a complete sentence, etc. However the paper also present more general methods, including a method for building a highly covering map from words to polarities based on a small set of positive and negative seed words. This method has been adopted by this approach as it solves the assignment of polarity values on the lexical level quite elegantly, and is very loosely coupled to the domain. However their actual semantic analysis, which unfortunately is described somewhat shallow in the paper, seem to suffer from severe problems with respect to certain phrase structures, e.g. *dependent clauses*.

Finally it is noted, that there seem to be a strong imbalance between the *formal approaches* and *machine learning approaches*, with respect to amount of research, i.e. there exists a lot of research on sentiment analysis using machine learning compared to research embracing formal methods.

3 Material and Methods

3.1 Combinatory Categorial Grammar

The grammar formalism used is *Combinatory Categorial Grammar* (CCG), pioneered largely by [18], and later enhanced by [1] to incorporate *modalities*. CCG adds a layer of combinatory logic onto pure Categorial Grammar, which allows an elegant and succinct formation of *higher-order* semantic expressions directly from the syntactic analysis. The set of modalities used, $\mathcal{M}$, follows [1] and [19], where $\mathcal{M} = \{\star, \diamond, \times, \cdot\}$. The set is partially ordered cf. the lattice (3).

$$\begin{tikzcd} & \star \arrow[dl, dash] \arrow[dr, dash] & \\ \diamond \arrow[dr, dash] & & \times \arrow[dl, dash] \\ & \cdot & \end{tikzcd} \tag{3}$$

A CCG lexicon, $\mathcal{L}_{\text{CCG}}$, is a mapping from a lexical unit, $w \in \Sigma^\star$, to a set of 2-tuples, each containing a lexical category and semantic expression that the unit can entail cf. (4), where Γ denotes the set of lexical and phrasal categories, and Λ denotes the set of semantic expressions.

$$\mathcal{L}_{\text{CCG}} : \Sigma^\star \to \mathcal{P}(\Gamma \times \Lambda) \tag{4}$$

A category is either *primitive* or *compound*. The set of primitive categories, $\Gamma_{\text{prim}} \subset \Gamma$, is language dependent and, for the English language, 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 α and β are members of Γ, then so are $\alpha/_\iota\beta$ and $\alpha\backslash_\iota\beta$. The modality of the operator, $\iota \in \mathcal{M}$, can restrict the application of inference rules during deduction in order to ensure the soundness of the system. 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 $\alpha/_.\beta$. Since $\cdot$ permits any rule it is convenient to simply write $/$ and $\backslash$ instead of respectively $/_.$ and $\backslash_.$, i.e. the dot is omitted from these operators.

3.2 Combinatory Rules

CCGs can be seen as a logical deductive proof system where the axioms are members of $\Gamma \times \Lambda$. A text $T \in \Sigma^\star$ is accepted as a sentence in the language, if there, for some tagging of T, exists a deductive proof for S (sentence). A tagging of a text is, for each lexical unit $w \in \Sigma^\star$ in the text, simply the selection of one of the pairs yielded by $\mathcal{L}_{\text{CCG}}(w)$. While this seems simple, it constitutes the major computational challenge of this approach. E.g. given some ordered set of lexical units, which constitutes the text T to analyse, the number of possible combinations of taggings might be very large.

Once an appropriate selection is made rewrite is done using a language independent set of *combinatory rules*:

$$\begin{array}{rrcll}
X/_{\star}Y : f & Y : a & \Rightarrow & X : f\ a & (>) \\
Y : a & X\backslash_{\star}Y : f & \Rightarrow & X : f\ a & (<) \\
X/_{\diamond}Y : f & Y/_{\diamond}Z : g & \Rightarrow & X/_{\diamond}Z : \lambda a.f(g\ a) & (>_{\mathbf{B}}) \\
Y\backslash_{\diamond}Z : g & X\backslash_{\diamond}Y : f & \Rightarrow & X\backslash_{\diamond}Z : \lambda a.f(g\ a) & (<_{\mathbf{B}}) \\
 & X : a & \Rightarrow & T/(T\backslash X) : \lambda f.fa & (>_{\mathbf{T}}) \\
 & X : a & \Rightarrow & T\backslash(T/X) : \lambda f.fa & (<_{\mathbf{T}}) \\
X/_{\times}Y : f & Y\backslash_{\times}Z : g & \Rightarrow & X\backslash_{\times}Z : \lambda a.f(g\ a) & (>_{\mathbf{B}_{\times}}) \\
Y/_{\times}Z : g & X\backslash_{\times}Y : f & \Rightarrow & X/_{\times}Z : \lambda a.f(g\ a) & (<_{\mathbf{B}_{\times}})
\end{array}$$

Where X, Y, Z and T are variables ranging over categories (i.e. Γ), and f, a and g are variables over semantic expressions (i.e. Λ).

With only functional application, $(>)$ and $(<)$, the system is capable of capturing any context-free language cf. [18]. Figure 1 shows the deduction of S from the simple declarative sentence "the hotel had an exceptional service" (semantics are omitted).

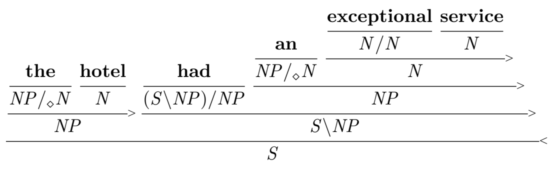


Fig. 1. Deduction of simple declarative sentence.

The functional composition, $(>_{\mathbf{B}})$ and $(<_{\mathbf{B}})$ is often used in connection with type-raising $(>_{\mathbf{T}})$ and $(<_{\mathbf{T}})$, for instance to allow relative clauses, coordination, while crossed functional composition, $(>_{\mathbf{B}_{\times}})$ and $(<_{\mathbf{B}_{\times}})$ are needed for more exotic linguistic phenomenons such as *heavy noun phrase shifting*.

3.3 Maximum Entropy Tagging

There exists some wide covering CCG lexicons, most notable *CCGbank*, compiled by [10] by techniques presented by [9]. It is essentially a translation of almost the entire Penn Treebank [12], which contains over 4.5 million lexical units, 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, Γ. The problem of obtaining semantic expressions, is addressed later.

To ensure the presented method works with a large vocabulary and a wide range of sentence structures, and thus the variety of opinion texts harvested from social networks, an efficient syntactical CCG-tagging is required. This is substantiated by the fact that [10] calculates that the expected number of lexical categories per token is 19.2 for the CCGBank. This mean that an exhaustive search of even a short sentence (seven tokens) is expected to consider over 960 million (19.2^7) possible taggings.

Machine learning is used to handle this otherwise exponentially bounded search, specifically [4] presents a method based on a *maximum entropy model* that estimates 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 lexical units, and the CCG-category of lexical units left to the current.

This is used to select a subset of possible categories for a lexical unit, by selecting categories with a probability within a factor of the category with highest probability. In some cases this of cause will prune the correct tagging needed to deduct S, but [4] shows that the average number of lexical categories per lexical unit can be reduced to 3.8 while the method still recognize 98.4% of unseen data.

A complete parser is presented in [5]. It utilizes this tagging model and a series of (log-linear) models to speed-up the actual deduction once the tagging model has assigned a set of categories to each token.

Finally, since the tagging models are based on trained data, which also can contain minor grammatical errors and misspellings, it is still able to assign categories to lexical entries even though they might be incorrect spelled or of wrong form, which it not very uncommon when harvesting data from social networks, user reviews, etc.

4 Theory and Calculation

4.1 Definition of a Sentiment Analysis

The sentiment polarity model used in this paper is continuous, and can thus be seen as a weighted classification. Thereby the polarity is a value in some predefined interval, $[-\omega; \omega]$. An opinion with value close to $-\omega$ is considered highly negative, whereas a value close to ω is considered highly positive. Opinions with values close to zero are considered almost neutral. This model allows the overall process of the sentiment analysis presented by this thesis to be given by Definition 1.

Definition 1. *A sentiment analysis $\mathcal{A}$ is a computation on a text $T \in \Sigma^\star$ with respect to a* subject of interest *$s \in \mathbb{E}$, where $\Sigma^\star$ denotes the set of all texts, and $\mathbb{E}$ is the set of all entities. The result is a normalized score as shown in (5). The yielded score should reflect the* polarity *of the given subject of interest in the text, i.e. whether the overall opinion is positive, negative, or neutral.*

$$\mathcal{A} : \Sigma^\star \to \mathbb{E} \to [-\omega; \omega] \tag{5}$$

4.2 Combinatory Categorial Grammar for Sentiment Analysis

In order to apply the CCG formalism to the area of sentiment analysis the expressive power of the semantics needs to be adapted to this task. The set of semantic expressions, Λ, is defined as a superset of simply typed λ-expressions cf. Definition 2.

Definition 2. *Besides variables, functional abstraction and functional application, which follows from simply typed λ-expressions cf. [2], 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 impact argument*.*
- *The* change *of an expression's polarity.*
- *The* scale *of an expression's polarity. The magnitude of which an expression's polarity may scale is given by $[-\psi; \psi]$.*

Formally this can be stated:

$$
\begin{array}{rcll}
x : \tau \in \mathcal{V} & \Rightarrow & x : \tau \in \Lambda & \text{(Variable)} \\
x : \tau_\alpha \in \mathcal{V},\ e : \tau_\beta \in \Lambda & \Rightarrow & \lambda x.e : \tau_\alpha \to \tau_\beta \in \Lambda & \\
 & & & \text{(Abstraction)} \\
e_1 : \tau_\alpha \to \tau_\beta \in \Lambda,\ e_2 : \tau_\alpha \in \Lambda & \Rightarrow & (e_1 e_2) : \tau_\beta \in \Lambda & \text{(Application)} \\
e_1, \ldots, e_n \in \Lambda, 0 \leq k \leq n,\ j \in [-\omega; \omega] & \Rightarrow & f_j^k(e_1, \ldots, e_n) \in \Lambda & \text{(Functor)} \\
e_1 : \tau, \ldots, e_n : \tau \in \Lambda & \Rightarrow & \langle e_1, \ldots, e_n \rangle : \tau \in \Lambda & \text{(Sequence)} \\
e : \tau \in \Lambda, 0 \leq k' & \Rightarrow & e^{\rightsquigarrow k'} : \tau & \text{(Impact change)} \\
e : \tau \in \Lambda,\ j \in [-\omega; \omega] & \Rightarrow & e_{\circ j} : \tau \in \Lambda & \text{(Change)} \\
e : \tau \in \Lambda,\ j \in [-\psi; \psi] & \Rightarrow & e_{\bullet j} : \tau \in \Lambda & \text{(Scale)}
\end{array}
$$

The semantics includes normal α-conversion and β-, η-reduction as shown in the semantic rewrite rules for the semantic expressions given by Definition 3. 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 applies 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 β-reduction rule. Completely analogous rules are provided for the scaling as shown in respectively (FS1), (FS2), (SS) and (PS). Finally the *change of impact* allows change of a functors impact argument cf. (IC). Notice that these *change*, *scale*, *push* and *impact change* rules are type preserving, and for readability type annotation is omitted from these rules.

Definition 3. *The rewrite rules of the semantic expressions are given by the following, 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, [2].*

$$\begin{array}{rcll}
(\lambda x.e) : \tau & \Rightarrow & (\lambda y.e[x \mapsto y]) : \tau & y \notin FV(e) \quad (\alpha) \\
((\lambda x.e_1) : \tau_\alpha \to \tau_\beta)\,(e_2 : \tau_\alpha) & \Rightarrow & e_1[x \mapsto e_2] : \tau_\beta & (\beta) \\
(\lambda x.(e\,x)) : \tau & \Rightarrow & e : \tau & x \notin FV(e) \quad (\eta) \\
f^0_j(e_1, \ldots, e_n)_{\circ j'} & \Rightarrow & f^0_{j \widehat{+} j'}(e_1, \ldots, e_n) & (\text{FC1}) \\
f^k_j(e_1, \ldots e_n)_{\circ j'} & \Rightarrow & f^k_j(e_1, \ldots, e_{k \circ j'}, \ldots e_n) & (\text{FC2}) \\
\langle e_1, \ldots, e_n \rangle_{\circ j'} & \Rightarrow & \langle e_{1 \circ j'}, \ldots, e_{n \circ j'} \rangle & (\text{SC}) \\
(\lambda x.e)_{\circ j'} & \Rightarrow & \lambda x.(e_{\circ j'}) & (\text{PC}) \\
\\
f^0_j(e_1, \ldots, e_n)_{\bullet j'} & \Rightarrow & f^0_{j \widehat{\cdot} j'}(e_1, \ldots, e_n) & (\text{FS1}) \\
f^k_j(e_1, \ldots e_n)_{\bullet j'} & \Rightarrow & f^k_j(e_1, \ldots, e_{k \bullet j'}, \ldots e_n) & (\text{FS2}) \\
\langle e_1, \ldots, e_n \rangle_{\bullet j'} & \Rightarrow & \langle e_{1 \bullet j'}, \ldots, e_{n \bullet j'} \rangle & (\text{SS}) \\
(\lambda x.e)_{\bullet j'} & \Rightarrow & \lambda x.(e_{\bullet j'}) & (\text{PS}) \\
\\
f^k_j(e_1, \ldots e_n)^{\leadsto k'} & \Rightarrow & f^{k'}_j(e_1, \ldots e_n) & (\text{IC})
\end{array}$$

It is assumed that the addition and multiplication operator, respectively $\widehat{+}$ and $\widehat{\cdot}$, always yields a result within $[-\omega; \omega]$ cf. Definition 4.

Definition 4. *The operators $\widehat{+}$ and $\widehat{\cdot}$ are defined cf. (6) and (7) such that they always yield a result in the range $[-\omega; \omega]$, even if the pure addition and multiplication might not be in this range.*

$$j \widehat{+} j' = \begin{cases} -\omega & \textit{if } j + j' < -\omega \\ \omega & \textit{if } j + j' > \omega \\ j + j' & \textit{otherwise} \end{cases} \tag{6}$$

$$j \widehat{\cdot} j' = \begin{cases} -\omega & \textit{if } j \cdot j' < -\omega \\ \omega & \textit{if } j \cdot j' > \omega \\ j \cdot j' & \textit{otherwise} \end{cases} \tag{7}$$

The presented definition of semantic expressions allows the binding between expressed sentiment and entities in the text to be analyzed, given that each lexicon entry have associated the proper expression.

Example 1 shows how to apply this for a simple declarative sentence, while Example 2 considers an example with long distance dependencies.

Example 1. This example considers simple declarative sentence (8) including semantics.

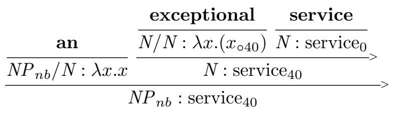
the hotel had an exceptional service (8)

The lexicon used in the example is provided in Table 1. Notice that nouns, verbs, etc. are reduced to their lemma for functor naming.

Token	Category		Type
the	NP_{nb}/N	:	$\lambda x.x$
hotel	N	:	hotel_0
had	$(S_{dcl}\backslash NP)/NP$	:	$\lambda x.\lambda y.\mathrm{have}_0^0(x, y)$
an	NP_{nb}/N	:	$\lambda x.x$
exceptional	N/N	:	$\lambda x.(x_{\circ 40})$
service	N	:	$\mathrm{service}_0$

Table 1. Lexicon used in Example 1.

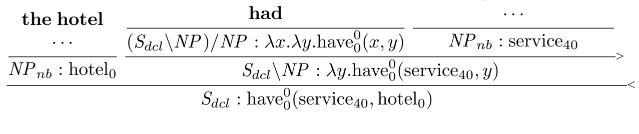
Figure 2 shows the entity "service" is modified by the adjective "exceptional" which is immediately to the left of the entity. The semantic expression associated to "service" is simply the zero-argument functor, initial with a neutral sentiment value. The adjective has the "changed identity function" as expression with a change value of 40. Upon application of combinatorial rules, semantic expressions are reduced based on the rewrite rules given in Definition 3.



an
exceptional
service
$NP_{nb}/N : \lambda x.x$
$N/N : \lambda x.(x_{\circ 40})$
$N : \mathrm{service}_0$
>
$N : \mathrm{service}_{40}$
>
$NP_{nb} : \mathrm{service}_{40}$

Fig. 2. Deduction of simple noun phrase sentence with semantics.

The conclusion of the deduction proof is a sentence with a semantic expression preserving most of the surface structure, and includes the bounded sentiment values on the functors cf. Figure 3.



the hotel
had
an exceptional service
$\cdots$
$\cdots$
$NP_{nb} : \mathrm{hotel}_0$
$(S_{dcl}\backslash NP)/NP : \lambda x.\lambda y.\mathrm{have}_0^0(x, y)$
$NP_{nb} : \mathrm{service}_{40}$
>
$S_{dcl}\backslash NP : \lambda y.\mathrm{have}_0^0(\mathrm{service}_{40}, y)$
<
$S_{dcl} : \mathrm{have}_0^0(\mathrm{service}_{40}, \mathrm{hotel}_0)$

Fig. 3. Deduction of simple declarative sentence with semantics.

Example 2. This example considers the sentence (9) including semantics, and demonstrates variations of all combinator rules introduced.

the breakfast that the restaurant served daily was excellent (9)

The lexicon used in the example is provided in Table 2. Most interesting is the correct binding between "breakfast" and "excellent", even though these are far from each other in the surface structure of the sentence.

Token	Category		Type
the	NP_{nb}/N	:	$\lambda x.x$
breakfast	N	:	breakfast_0
that	$(N\backslash_\diamond N)/(S_{dcl}/_\diamond NP)$	:	$\lambda x.\lambda y.((x\ y)^{\leadsto 1})$
restaurant	N	:	restaurant_0
served	$(S_{dcl}\backslash NP)/NP$	:	$\lambda x.\lambda y.\text{serve}_0^0(x,y)$
daily	$(S_X\backslash NP)\backslash(S_X\backslash NP)$	:	$\lambda x.(x_{\circ 5})$
was	$(S_{dcl}\backslash NP)/(S_{adj}\backslash NP)$	:	$\lambda x.x$
excellent	$S_{dcl}\backslash NP$	:	$\lambda x.(x_{\circ 25})$

Table 2. Lexicon used in Example 2.

Figure 4 shows how the adverb "daily" correctly modifies the transitive verb "served", even though the verb is missing it's object since it participates in a relative clause.

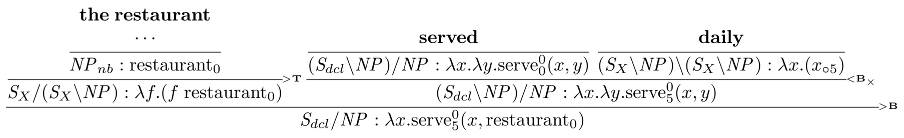

the restaurant
served
daily
$NP_{nb} : \text{restaurant}_0$
$(S_{dcl}\backslash NP)/NP : \lambda x.\lambda y.\text{serve}_0^0(x,y)$
$(S_X\backslash NP)\backslash(S_X\backslash NP) : \lambda x.(x_{\circ 5})$
$>\mathbf{T}$
$<\mathbf{B}_\times$
$S_X/(S_X\backslash NP) : \lambda f.(f\ \text{restaurant}_0)$
$(S_{dcl}\backslash NP)/NP : \lambda x.\lambda y.\text{serve}_5^0(x,y)$
$>\mathbf{B}$
$S_{dcl}/NP : \lambda x.\text{serve}_5^0(x,\text{restaurant}_0)$

Fig. 4. Deduction of dependent clause.

Figure 5 shows the details of the relative clause. When the relative pronoun binds the dependent clause to the main clause, it "closes" it for further modification by changing the impact argument of the functor inflicted by the verb of the dependent clause, so that further modification will impact the subject of the main clause.

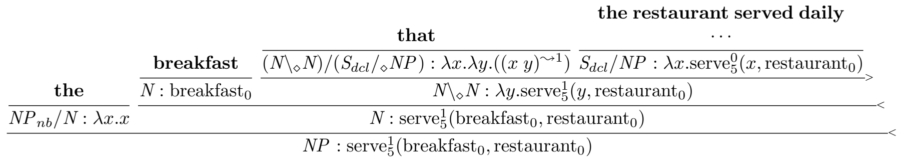

the restaurant served daily
that
breakfast
the
$(N\backslash_\diamond N)/(S_{dcl}/_\diamond NP) : \lambda x.\lambda y.((x\ y)^{\leadsto 1})$
$S_{dcl}/NP : \lambda x.\text{serve}_5^0(x,\text{restaurant}_0)$
$N : \text{breakfast}_0$
$N\backslash_\diamond N : \lambda y.\text{serve}_5^1(y,\text{restaurant}_0)$
$NP_{nb}/N : \lambda x.x$
$N : \text{serve}_5^1(\text{breakfast}_0,\text{restaurant}_0)$
$NP : \text{serve}_5^1(\text{breakfast}_0,\text{restaurant}_0)$

Fig. 5. Binding of relative clause and noun phrase.

Finally the the long distance binding can be etablished as shown in Figure 6.

$$
\begin{array}{ccc}
\textbf{the breakfast that the restaurant served daily} & \textbf{was} & \textbf{excellent} \\
 & \overline{(S_{dcl}\backslash NP)/(S_{adj}\backslash NP) : \lambda x.x} & \overline{S_{adj}\backslash NP : \lambda x.(x_{\circ 25})} \\
\cdots & \multicolumn{2}{c}{} \\
\hline
NP : \text{serve}_5^1(\text{breakfast}_0, \text{restaurant}_0) & \multicolumn{2}{c}{\overline{S_{dcl}\backslash NP : \lambda x.(x_{\circ 25})}\,>} \\
\hline
\multicolumn{3}{c}{S_{dcl} : \text{serve}_5^1(\text{breakfast}_{25}, \text{restaurant}_0) \quad <}
\end{array}
$$

Fig. 6. Sentiment of sentence with long distance dependencies.

4.3 Lexicon Annotation

There is however one essential component missing from the lexicon, namely the semantic expressions. However due to the *Principle of Categorial Type Transparency* it is known exactly *what* the types of the semantic expressions should be. There are currently a total of 429 different tags in the *maximum entropy tagging model*, thus trying to handle each of these cases individually is certainly not very robust for changes in the lexical categories. The solution is to handle some cases that need special treatment, and then use a generic annotation algorithm for all other cases. Both the generic and the special case algorithms will be a transformation $(\mathcal{T}, \Sigma^\star) \to \Lambda$, where the first argument is the type, $\tau \in \mathcal{T}$, to construct, and the second argument is the lemma, $\ell \in \Sigma^\star$, of the lexicon entry to annotate. Since the special case algorithms will fallback to the generic approach, in case preconditions for the case are not met, it is convenient to start with the generic algorithm, $\mathcal{U}_{\text{GEN}}$, which is given by Definition 5.

Definition 5. *The* generic semantic annotation algorithm*, $\mathcal{U}_{\text{GEN}}$ (10), for a type τ and lemma ℓ is defined by the auxiliary function $\mathcal{U}'_{\text{GEN}}$, which takes two additional arguments, namely an infinite set of variables $\mathcal{V}$ cf. Definition 2, and an ordered set of sub-expressions, (denoted A), which initially is empty.*

$$\mathcal{U}_{\text{GEN}}(\tau, \ell) = \mathcal{U}'_{\text{GEN}}(\tau, \ell, \mathcal{V}, \emptyset) \tag{10}$$

If τ is primitive, i.e. $\tau \in \mathcal{T}_{\text{prim}}$, then the generic algorithm simply return a functor with name ℓ, polarity and impact argument both set to 0*, and the ordered set A as arguments. Otherwise there must exist unique values for $\tau_\alpha, \tau_\beta \in \mathcal{T}$, such that $\tau_\alpha \to \tau_\beta = \tau$, and in this case the algorithm return an abstraction of τ_α on variable $v \in V$, and recursively generates an expression for τ_β.*

$$\mathcal{U}'_{\text{GEN}}(\tau, \ell, V, A) = \begin{cases} \ell_0^0(A) : \tau & \textit{if } \tau \in \mathcal{T}_{\text{prim}} \\ \lambda v.\mathcal{U}'_{\text{GEN}}(\tau_\beta, \ell, V \setminus \{v\}, A') : \tau & \textit{otherwise, where:} \end{cases}$$

$$
\begin{aligned}
v &\in V \\
\tau_\alpha \to \tau_\beta &= \tau \\
A' &= \begin{cases} A[e : \tau_\alpha \to \tau_\gamma \mapsto ev : \tau_\gamma] & \textit{if } e' : \tau_\alpha \to \tau_\gamma \in A \\ A[e : \tau_\gamma \mapsto ve : \tau_\delta] & \textit{if } \tau_\gamma \to \tau_\delta = \tau_\alpha \wedge e' : \tau_\gamma \in A \\ A \cup \{v : \tau\} & \textit{otherwise} \end{cases}
\end{aligned}
$$

The recursive call also removes the abstracted variable v from the set of variables, thus avoiding recursive abstractions to use it. The ordered set of sub-expressions, A, is modified cf. A', where the notation $A[e_1 : \tau_1 \mapsto e_2 : \tau_2]$ is the substitution of all elements in A of type τ_1 with $e_2 : \tau_2$. Note that e_1 and τ_1 might be used to determine the new value and type of the substituted elements. Since the two conditions on A' are not mutual exclusive, if both apply the the first case will be selected. The value of A' can be explained in an informal, but possibly easier to understand, manner:

– *If there is a least one function in A, that takes an argument of type τ_α, then apply v (which is known to by of type τ_α) to all such functions in A.*

– *If the type of v itself is a function (i.e. $\tau_\gamma \rightarrow \tau_\delta = \tau_\alpha$), and A contains at least one element that can be used as argument, then substitute all such arguments in A by applying them to v.*

– *Otherwise, simply append v to A.*

Clearly the generic algorithm does not provide much use with respect to extracting the sentiment of entities in the text, i.e. it only provide some safe structures that are guaranteed to have the correct type. The more interesting annotation is actually handled by the special case algorithms. How this is done is determined by a combination of the POS-tag and the category of the entry. Most of these treatments are very simple, with the handling of adjectives and adverbs being the most interesting:

– *Determiners* with simple category, i.e. NP/N, are simply mapped to the identity function, $\lambda x.x$. While determiners have high focus in other NLP tasks, such as determine if a sentence is valid, the importance does not seem significant in sentiment analysis, e.g. whether an opinion is stated about *an entity* or *the entity* does not change the overall polarity of the opinion bound to that entity.

– *Nouns* are in general just handled by the generic algorithm, however in some cases of multi-word nouns, the sub-lexical entities may be tagged with the category N/N. In these cases the partial noun is annotated with a list structure, that eventually will capture the entire noun, i.e. $\lambda x.\langle \mathcal{U}_{\text{GEN}}(\tau_{\text{N}}, \ell), x\rangle$, where ℓ is the lemma of the entity to annotate.

– *Verbs* are just as nouns in general handled by the generic algorithm, however *linking verbs* is a special case, since they relate the subject (i.e. an entity) with one or more *predicative adjectives*. Linking verbs have the category $(S_{\text{dcl}}\backslash NP)/(S_{\text{adj}}\backslash NP)$, and since the linked adjectives directly describes the subject of the phrase such verbs are simply annotated with the identity function, $\lambda x.x$.

– *Adjectives* can have a series of different categories depending on how they participate in the sentence, however most of them have the type $\tau_\alpha \rightarrow \tau_\beta$, where $\tau_\alpha, \tau_\beta \in \mathcal{T}_{\text{prim}}$. These are annotated with the *change* of the argument, i.e. $\lambda x.x_{\circ j}$, where j is a value determined based on the lemma of the adjective. Notice that this assumes implicit type conversion of the parameter from τ_α to τ_β, however since these are both primitive, this is a sane type cast. Details on how the value j is calculated are given in the next section.

– *Adverbs* are annotated in a fashion closely related to that of adjectives. However the result might either by a *change* or a *scale*, a choice determined by the lemma: normally adverbs are annotated by the change in the same manner as adjectives, however *intensifiers* and *qualifiers*, i.e. adverbs that respectively strengthens or weakens the meaning, are scaled. The next section gives further details on how this choice is made. Finally special care are taken about negating adverbs, i.e. "not", which are scaled with a value $j = -1$.
– *Prepositions* and *relative pronouns* need to change the impact argument of captured partial sentences, i.e. *preposition phrases* and *relative clauses*, such that further modification should bind to the subject of the entire phrase as were illustrated by Example 2.
– *Conjunctions* are annotated by an algorithm closely similar to $\mathcal{U}_{\text{GEN}}$, however instead of yielding a functor of arguments, the algorithm yields a list structure. This allow any modification to bind on each of the conjugated sub-phrases.

4.4 Assignment of Sentiment Polarity

An understanding of the domain of the review is needed in order to reason about the polarity of the entities present in texts to analyse. For this purpose the concept of *semantic networks* was used. Concretely the semantic network WordNet, originally presented by [13], and later presented in depth by [7]. WordNet contains a variety of relations, however for the purpose of calculating sentiment polarity values, only the following were considered interesting:

– The *similar*-relation, r_{similar}, links *closely similar* semantic concepts, i.e. concepts having almost the synonymies mensing in most contexts. The relation is present for most concepts entailed by adjectives.
– The *see also*-relation, $r_{\text{see-also}}$, links *coarsely similar* semantic concepts, i.e. concepts having a clear different meaning, but may be interpreted interchangeably for some contexts.
– The *pertainym*-relation, $r_{\text{pertainym}}$, links the adjectives from which an adverb was derived, e.g. *extreme* is the pertainym of *extremely.*

An approach similar to the one presented by [17] was used to calculate an assignment of sentiment polarity values for adjectives and adverbs: Positive and negative *seed concepts* are identified for the domain of the analysis, respectively S_{pos} and S_{neg}, e.g. as shown in (11) and (12).

$$S_{\text{pos}} = \{\text{clean}, \text{quiet}, \text{friendly}, \text{cheap}\} \tag{11}$$
$$S_{\text{neg}} = \{\text{dirty}, \text{noisy}, \text{unfriendly}, \text{expensive}\} \tag{12}$$

The calculation of the polarity *change* and/or *scale* for some *lemma*, present in the texts to analyze, is then based on the distances between the concepts yielded by the lemma and the seed concepts. To solve semantic ambiguity a rational assumption was taken that concepts stated in the texts presumably are to be interpreted within the domain given by the seed concepts, S_{pos} and S_{neg}. Thus concepts that are strongly related to one or more seed concepts should be preferred over weaklier related concepts. The solution is to select the *n closest* relations, thus reasoning greedily positively, respectively greedily negatively.

The approach for calculating sentiment polarity values for *intensifying* or *qualifying* adverbs modify the meaning of a verb, adjective, or another adverb, some special treatment are presented for this. Analog to the positive and negative concepts, sets of respectively intensifying and qualifying seed adjectives are stated, e.g. (13) and (14). Also notice that, unlike S_{pos} and S_{neg}, these sets does not rely on the domain, as they only strengthens or weakens domain specific polarities.

$$S_{\text{intensify}} = \{\text{extreme}, \text{much}, \text{more})\} \tag{13}$$

$$S_{\text{qualify}} = \{\text{moderate}, \text{little}, \text{less})\} \tag{14}$$

The distances are normalized such that the change value for some adverb or adjective with lemma, ℓ, will always be in $[-\omega; \omega]$; and for intensifying or qualifying adverbs with lemma, ℓ, the scale will always be in $\left[\frac{1}{2}; 2\right]$.

5 Results

5.1 Test Data

The test data set chosen for evaluation of the method was the *Opinosis data set* [8]. The data set consists of approximately 7000 texts from consumer reviews on a number of different topics. The topics are ranging over different product and services, from consumer electronics (e.g. GPS navigation, music players, etc.) to hotels and restaurants. They are harvested from several online resellers and service providers, including among others *Amazon* (`http://www.amazon.com/`) and *TripAdvisor* (`http://www.tripadvisor.com/`).

Since the data set is unlabeled it was chosen to label a small subset of it in order to measure the robustness and the correctness of the presented method (see Appendix). To avoid biases toward how the proof of concept system analyzes text the labeling was performed independently by two individuals who had no knowledge of how the presented solution processes texts.

As the example texts might have hinted, the subset chosen was from the set of hotel and restaurant reviews. The *subject of interest* chosen for the analysis were *hotel rooms*, and the subset was thus randomly sampled from texts with high probability of containing this entity (i.e. containing any morphological form of the noun "room").

The individuals were given a subset of 35 review texts, and should mark each text as either positive, negative or unknown *with respect to the given subject of* *interest*. Out of the 35 review text the two subject's positive/negative labeling agreed on 34 of them, while unknowns and disagreements were discarded. Thus the inter-human *concordance* for the test data set was 97.1%, which is very high, and would arguably drop if just a few more individuals were used for label annotation.

5.2 Evaluation Results

An entity sentiment value was considered to *agree* with the human labeling, if it had the correct sign (i.e. positive sentiment values agreed with positive labels, and negative values with negative labels). The baseline presented here is a sentence-level baseline, calculated by using the Naive Bayes Classifier available in the Natural Language Toolkit (NLTK) for Python.

The *precision* and *recall* results for both the baseline, and the presented method are shown in Table 3. As seen the recall is somewhat low for the proof of concept system, which is addressed in the next section, while it is argued that precision of the system is indeed acceptable, since even humans will not reach a 100% agreement.

	Baseline	The presented method
Precision	71.5%	92.3%
Recall	44.1%	35.3%

Table 3. Precision and recall results for proof of concept system.

6 Discussion

The presented method for *entity level* sentiment analysis using deep sentence structure analysis has shown acceptable correction results, but inadequate robustness.

The biggest issue was found to be the lack of correct syntactic tagging models. It is argued that models following a closer probability distribution of review texts than the one used would have improved the robustness of the system significantly. One might think, that if syntactic labeled target data are needed, then the presented logical method really suffers the same issue as machine learning approaches, i.e. *domain dependence*. However it is argued that exactly because the models needed are of *syntactic level*, and not of *sentiment level*, they really do not need to be *domain specific*, but only *genre specific*. This reduces the number of models needed, as a syntactic tagging model for reviews might cover several domains, and thus the *domain independence* of the presented method is intact.

Especially the property of being domain independent is considered to be of significant importance of the presented method. As harvested data grows, and new domains surfaces (e.g. *Internet of Things*) any method requiring labeled training data will be slower and more costly to deploy. Besides the savings of avoiding expensive computational training of domain specific models, the method also allows the ability to *reuse* models on new and unseen domains, as long as some *domain expert* provides the seed concepts (S_{pos} and S_{neg}).

An interesting experiment would have been to see how the presented method performed on such genre specific syntactic models. [16] presents methods for *cross-domain semi-supervised learning*, i.e. the combination of labeled (e.g. *CCG-Bank*) and unlabeled (e.g. review texts) data from different domains (e.g. syntactic genre). This allows the construction of models that utilizes the knowledge present in the labeled data, but also biases it toward the distribution of the unlabeled data. The learning accuracy is of cause not as significant as compared to learning with large amounts of labeled target data, but it can improve genres where no labeled data are available.

7 Conclusions

This paper has presented a *formal logical method* for *entity level* sentiment analysis, which utilizes *machine learning techniques* for efficient syntactic tagging. The method should be seen as an alternative to pure machine learning methods, which have been argued inadequate for capturing long distance dependencies between an entity and opinions, and of being highly dependent on the domain of the sentiment analysis.

Empirical results showed that while the correctness of the presented method seems acceptably high, its robustness is currently inadequate for most real-world applications. However, it is argued that it indeed is possible to improve the robustness significantly with further refinements of the presented method.

Besides resolving the issue of the low robustness, the presented method also leaves plenty of opportunities for expansion. This could include a more sophisticated pronoun resolution, and even more advanced extraction strategies could also include relating entities by the use of some of the abstract topological relations available in semantic networks. E.g. *hyponym*/*hypernym* and *holonym*/*meronym*. With such relations, a strong sentiment of the entity *room* might inflict the sentiment value of *hotel*, since *room* is a meronym of *building*, and *hotel* is a hyponym of *building*.

In the future we aim to use advanced mathematical proof assistants like *Coq* (`https://coq.inria.fr/`) and *Isabelle* (`https://isabelle.in.tum.de/`) for the formalization of the presented theory. The proof assistants have support for the necessary data structures and algorithms. The use of proof assistants would allow for formal proofs of key properties and also for easier experiments with the presented method.

Appendix: Labeled Test Data

The following table consists of a random sample chosen from the "Swissotel Hotel" topic of the *Opinosis data set* [8] which contain any morphological form of the *subject of interest: hotel rooms*. Each sentence in the data set (which may not constitute a complete review) has been labeled independently by two human individuals *with respect to the subject of interest: hotel rooms*. Furthermore the table contains results for the presented method (entity level polarity value of subject of interest).

#	Review text	Humans	Method
1	The rooms are in pretty shabby condition, but they are clean.	Negative	Unknown
2	The rooms are spacious and have nice views, I was NOT impressed with the mattress and every, little, tiny thing costs money.	Unknown	N/A
3	The rooms look like they were just remodled and upgraded, there was an HD TV and a nice iHome docking station to put my iPod so I could set the alarm to wake up with my music instead of the radio.	Positive	Unknown
4	The rooms were cleaned spic and span every day.	Positive	Unknown
5	When I got to the room, I thought the new rooms would have a plasma since the website implies the new rooms would have them, but I guess those come later.	Negative	Unknown
6	Very impressed with rooms and view!	Positive	Unknown
7	The rooms are not all that big.	Negative	Unknown
8	Expensive Parking but great rooms.	Positive	30.0
9	Rooms were nicely furnished.	Positive	Unknown
10	The rooms are very clean, comfortable and spacious and up-to-date.	Positive	52.0
11	I've olny ever stayed in the "standard" rooms in this property, all of which are spacious and airy, and function well for both business or leisure travellers.	Positive	Unknown
12	It does suffer, however, from a trend that I have been noticing that as rooms at business class hotels are upgraded, particularly with a patch panel for the big LCD, TV, drawer space becomes less and less.	Negative	Unknown
13	We even got upgraded to one of the corner rooms which also looked west toward Michigan Ave and the Wrigley building.	Positive	Unknown
14	The rooms were very clean, the service was polite and helpful, and it's near the heart of Chicago!	Positive	52.0
15	You can see downtown and or the Navy Pier from most of the rooms.	Positive	Unknown

#	Review text	Humans	Method
16	no more bathrobes in corner rooms suites, coffee service in room is parred way down, the buffet offered in the cafe is not as bountiful, although the cafe staff is inpeccable and extremely gracious and will bring you what you wish, check in staff not at all eager to upgrade you, even though you may be a frequent visitor.	Negative	Unknown
17	Our rooms were nice and didn't look worn or old.	Positive	Unknown
18	Rooms at the hotel are getting somewhat tired.	Negative	0.8
19	Great Location great rooms and bed but no help from desk personnel.	Positive	38.0
20	While the rooms are quite nice, I was dismayed by the snotty service I received at the Swissotel in Chicago.	Positive	72.0
21	Rooms are dated, our corner room's bathroom was shabby.	Negative	Unknown
22	The hotel was very nice, rooms were big, the pool hot tub area was very nice, and the location was great and easy to get to.	Positive	10.0
23	Rooms are good quality and clean, what you would expect from a four star business hotel.	Positive	46.0
24	The view from the rooms was fantastic, My daughters are allergic to feathers and all trace of them were removed from the room as soon as we advised housekeeping.	Positive	Unknown
25	The Swissotel is one of our favorite hotels in Chicago and the corner rooms have the most fantastic views in the city.	Positive	Unknown
26	Then again, the rooms are much larger and the view more than makes up for it.	Positive	26.0
27	Rooms in similar hotels would usually be about $250, 300.	Positive	Unknown
28	The actual hotel and rooms were very nice with amazing views, the staff was extremely rude.	Positive	8.0
29	The rooms were clean, and upscale for the low price we paid.	Positive	Unknown
30	Thanks to TravelZoo I was able to find an amazing deal, lakeside rooms for $129 night as part of a spring promotion.	Positive	Unknown
31	I recieved a great deal on the rooms here and it was wonderful.	Positive	8.0
32	The room was huge as hotel rooms go.	Positive	26.0
33	Hotel was very clean and the rooms were comfy.	Positive	Unknown
34	word to the wise, avoid the rooms ending with 11.	Negative	Unknown
35	The rooms are large and well, appointed, the staff was very professional and friendly, and the view was striking!	Positive	34.0

References

1. Baldridge, J., Kruijff, G.J.M.: Multi-modal combinatory categorial grammar. In: Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics - Volume 1. pp. 211–218. EACL '03 (2003)
2. Barendregt, H., Dekkers, W., Statman, R.: Lambda Calculus with Types. Cambridge University Press (2013)
3. Cambria, E., Schuller, B., Liu, B., Wang, H., Havasi, C.: Statistical approaches to concept-level sentiment analysis. IEEE Intelligent Systems 28(3), 6–9 (2013)
4. Clark, S.: A supertagger for combinatory categorial grammar. In: International Workshop on Tree Adjoining Grammars and Related Frameworks. pp. 19–24. Venice, Italy (2002)
5. Clark, S., Curran, J.R.: Wide-coverage efficient statistical parsing with CCG and log-linear models. Computational Linguistics 33(4), 493–552 (2007)
6. Feldman, R.: Techniques and applications for sentiment analysis. Commun. ACM 56(4), 82–89 (Apr 2013)
7. Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database (Language, Speech, and Communication). The MIT Press, illustrated edition edn. (1998)
8. Ganesan, K., Zhai, C.X., Han, J.: Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. In: International Conference on Computational Linguistics. pp. 340–348 (2010)
9. Hockenmaier, J.: Data and Models for Statistical Parsing with Combinatory Categorial Grammar. Ph.D. thesis, University of Edinburgh (2003)
10. Hockenmaier, J., Steedman, M.: CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn treebank. Computational Linguistics 33(3), 355–396 (2007)
11. Liu, B.: Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data. Springer (2007)
12. Marcus, M.P., Marcinkiewicz, M.A., Santorini, B.: Building a large annotated corpus of English: The Penn treebank. Computational Linguistics 19(2), 313–330 (1993)
13. Miller, G.A.: WordNet: A lexical database for English. Communications of the ACM 38(11), 39–41 (1995)
14. Pang, B., Lee, L.: Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval 2(1-2), 1–135 (2008)
15. Petersen, N.C., Villadsen, J.: Combining formal logic and machine learning for sentiment analysis. In: Foundations of Intelligent Systems - 21st International Symposium, ISMIS 2014, Roskilde, Denmark, June 25-27, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8502, pp. 375–384. Springer (2014)
16. Søgaard, A.: Semi-supervised learning. ESSLLI-2012 Lecture (2012)
17. Simančík, F., Lee, M.: A CCG-based system for valence shifting for sentiment analysis. Research in Computing Science 41, 99–108 (2009)
18. Steedman, M.: The Syntactic Process. The MIT Press (2000)
19. Steedman, M.: Taking Scope: The Natural Semantics of Quantifiers. The MIT Press (2011)
20. Tan, L.K.W., Na, J.C., Theng, Y.L., Chang, K.: Sentence-level sentiment polarity classification using a linguistic approach. In: Proceedings of the 13th International Conference on Asia-pacific Digital Libraries: For Cultural Heritage, Knowledge Dissemination, and Future Creation. pp. 77–87. ICADL'11, Springer-Verlag (2011)