Recognising Textual Entailment

Marthe Catharina Dekker
9940715

Submitted in partial fulfilment of the requirements
for the degree of Master of Arts
Faculty of Arts
Utrecht University

Supervisors:
prof.dr. M.J. Moortgat
dr. W. Vermaat

2006

Contents

List of Figures

List of Tables

Acknowledgements

Thanks are due first of all to my supervisor Michael Moortgat, for his infectious enthusiasm, his support and his confidence.

Many thanks also go to my supervisor Willemijn Vermaat, for all discussions, guidance in the process of writing and hospitality. I could not have done it without her.

A collective word of thanks goes to the people at Utrecht University, Saarland University and elsewhere that have shown interest in my project and all authors that responded to questions or gave me access to their tools. I am grateful to UiL-OTS for sponsoring my visits to the ESSLLI summer schools.

I thank Antal van den Bosch for suggesting the RTE Challenge as a thesis subject.

I am very grateful to my friends and family and especially to my beloved one Ferdinand for their support, encouragement and love. Eric, thank you so much for the many discussions and help.

Chapter 1

Introduction

In modern natural language processing (NLP) applications, a broad variety of models of natural language are being used, from statistical models to highly sophisticated linguistic models, which are often based on unification grammars. Most NLP applications contain what we could identify as an interpretation mechanism and perform inferences. Two prominent approaches are the machine learning approach (usually with ad hoc, application-driven semantical representations), and the computational semantics approach, based on formal (model theoretic) semantics. There is no consensus about what meaning is and what a semantic representation should look like in a computational setting. Recently, work is also done on hybrid systems, combining methods from both approaches.

In NLP, we see a movement towards the development of more robust models with a wider coverage of naturally occurring language, in machine learning as well as in computational semantics [CMU+97]. Models are also more and more crossing the line between semantics and pragmatics. Blackburn and Bos [BB05] believe work in this direction to be very promising and also important for the development of semantics in general. We also see a trend towards models with a higher efficiency and lower costs.

For the evaluation of NLP systems, (natural language) test suites and corpora are developed, metrics are standardised and evaluation tasks are designed. Application challenges, like the MUC challenge in message understanding, have a stimulating effect on this development.

The evaluation of the wide coverage semantic and inferential performance of NLP applications can be done by running them against a new generic task called Recognizing Textual Entailment (RTE); the recognition of the degree in which one short natural language text follows from another one in a pragmatic way.

In this thesis, we design a model for the automatic recognition of Textual Entailment between short English texts, and evaluate it against the RTE task. Our design consists of semantic role extraction based on the output of a statistical parser and the use of a

common sense database as a measure for lexical similarity. Because of the explorative nature of the task, we compare the performance of this model to a shallow word overlap model and a somewhat linguistically enriched word overlap model. We are interested to see how well our design performs against the more shallow models and if the development data are usable for performing learning.

The structure of this thesis is as follows. Chapter 2 describes the RTE task and some characteristics of the data that we use for development. Chapter 3 gives an overview of our design. The two major modules, information extraction and use of a common sense database are described in Chapter 4 and Chapter 5 respectively. The performance of the design and of the two word overlap models are evaluated in Chapter 6. Chapter 7 contains conclusions and a short discussion.

Chapter 2

The RTE Task

2.1 Introduction

This chapter describes the Recognising Textual Entailment task (hereafter referred to as RTE) as a measurement for the semantic capacities of an interpretation mechanism. After motivating the use of inference tasks in Section 2.2, we give a detailed description of the RTE task in Section 2.3. In Section 2.4, we explore the linguistic characteristics of the corpus that we use for the development of our design.

2.2 The Semantic Competence of NLP Systems

The extraordinary expressivity of natural language is a big challenge for various natural language processing (NLP) applications. Most of the approaches to common and popular NLP tasks like Information Retrieval and Question Answering use some sort of semantic interpretation. The interpretation strategies that are used are various; deep semantic analysis, non-linguistic statistical methods, parse tree- or keyword matching, etcetera. These strategies are often application-driven: suited to the task at hand, but not generally usable for semantic interpretation in an arbitrary NLP system[1]. It is therefore difficult to compare them to one another and evaluate their semantic competence. An example of such a model that would need an evaluation measure for calculating semantic accuracy and -adequacy is the wide-coverage interpretation system described in [Bos05], which converts raw text into logical form and performs inferences with the results.

Performing an inference task is the best way of testing the semantic capacity of an NLP system; testing it against a set of inference schemata that represent most of the semantic phenomena that an interpretation system should be able to deal with. What these

[1]Other so-called *generic* tasks recognised within NLP are, for example, Word Sense Disambiguation, Named Entity Recognition and PoS-tagging.

phenomena are, is subject of discussion, but an attempt has been made by the FRACAS consortium [Con96]. They give a list of nearly 350 semantic inferences that could be the basis of a semantic test suite. It is a list of complex semantic phenomena like the ones we find in semantics handbooks, and by no means a representation of the real-world situation.

There was a need for a new generic and well-defined task that could be used to evaluate interpretation systems. In 2004, the PASCAL-network [2] organised a first challenge in Recognising Textual Entailment as a new task and evaluation framework for natural language understanding.

PASCAL is a world-wide network of researchers in natural language processing and statistical modelling, who are investigating and developing methods for the improvement of human-machine communication. PASCAL is coordinated by John Shawe-Taylor at the University of Southampton and organises application- as well as theoretical challenges in order to raise interest in and stimulate literature on these topics. Software challenges, in which software is evaluated by running it for a well-defined task, are an excellent possibility for scientists to test the quality of their software. They can turn out to be a catalyst for both research and application development, because the community gains insight in the state-of-the-art as well as the complexity of the task at hand.

RTE is a generic textual inference task which encloses a range of inferences which are relevant for and used in different NLP applications, like Information Retrieval, finding Comparable Documents, Reading Comprehension, Question Answering, Information Extraction, Machine Translation and Paraphrase Acquisition. The goal of the RTE challenge has been to raise interest in semantic interpretation by introducing a task a bit more simple than overall semantic interpretation and to evaluate how well different methods (existing, new and hybrid) would deal with this new task.

2.3 The RTE Task

The RTE task is to develop a strategy for automatically recognising if, for a pair of two short English newspaper texts (texts of one or two sentences), the first textually entails the second (*true*) or not (*false*). Additionally, for each pair, a confidence value must be given, ranging from 0 to 1, indicating the confidence that the system has in its own truth judgement, 1 indicating absolute confidence. Textual entailment is very different from semantic entailment. Let us compare the definitions:

Recognising Textual Entailment is the task of deciding if two unseen texts are in the Textual Entailment relation and giving a confidence score to this judgement.

[2] "PASCAL" stands for "Pattern Analysis, Statistical Modelling and Computational Learning". See http://www.pascal-network.org/ for more information about the network and http://www.pascal-network.org/Challenges/RTE/ for more information about this challenge, papers and data.

Textual Entailment (TE) is the relation between two texts for which holds that the truth of the former entails the truth of the latter.

In formal semantics, the definition of the term *entailment* is the following:

Semantic entailment : let $\Gamma = \{\phi_1, \ldots \phi_n\}$ be a set of sentences and ψ be a sentence. $\Gamma \models \psi$ (Γ semantically entails ψ) if for every model holds that $\phi_1 \wedge \ldots \wedge \phi_n$ = 0 or ψ = 1.

Note that semantic entailment is different from logical entailment in that the entailed sentence does not have to be *derived* from the entailing set.

Where semantic entailment is true or false without any doubt, textual entailment comes with a confidence score that can also be seen as indicating the level of entailment. Moreover, in contrast to semantic entailment, one can argue about the existence of textual entailment between two texts. In Chapter 7, we refer to a discussion in literature about the definition of textual entailment, the usefulness of RTE and the quality of the distributed data set.

Example 1 shows a pair of two sentences that are, according to a human annotator, in the textual entailment relation. A pair like this is called a *T-H pair*, its first sentence *Text* or *T*, the second one *Hypothesis* or *H*. We recognise in this example some of the peculiarities of the news domain. For example, in 1b, the weakening modal "may" from 1a has disappeared; it does not say that "Fiber may improve blood sugar control". Secondly, much is left unsaid in 1b, for example that you have to EAT LOTS OF fiber before some blood sugar control MAY follow, which makes 1b a typical headline. A good headline should follow from the text above which it stands and summarise it, but it is also short and powerful. Another example of a headline in the development data was the elliptical structure "No Weapons of Mass Destruction Found in Iraq Yet".

(1) a. *T*: Eating lots of foods that are a good source of fiber may keep your blood glucose from rising too fast after you eat.

b. *H*: Fiber improves blood sugar control.

2.4 The Development Data

A development corpus as well as an evaluation corpus are distributed with the task, containing about 570 and 800 *T-H pairs* respectively, xml-coded and manually annotated for existence of the entailment relation (*true/false*). Moreover, they are annotated as belonging to one of seven subsets, referring to NLP tasks that the pair would be a typical success- or failure example of. These subsets are: *IR* (Information Retrieval), *CD* (Comparable Documents), *RC* (Reading Comprehension), *QA* (Question Answering), *IE* (Information Extraction), *MT* (Machine Translation) and *PP* (Paraphrase Acquisition). An example of

an *MT* pair can be found in Example 2. It is annotated as *true*. *T* and *H* are an automatic translation and a gold-standard human translation of the same original sentence. The main characteristic of *MT* pairs is their parallelism on the word order level, whereas structurally they are rather different. The example illustrates some of the problems we are facing when trying to recognise entailment.

(2) a. *T*: An Iraqi official reported today, Saturday, that 68 Iraqi civilians were killed today as a result of the American and British bombing on Iraq and that their funerals were held today in Baghdad.

b. *H*: Reports from an Iraqi official today, Saturday, said that the 68 civilians killed as a result of the Anglo-American bombing of Iraq, were buried today in Baghdad.

Within the pairs annotated for each of the seven subsets, there are examples of various inference types (e.g. syntactic, semantic and pragmatic) and various difficulty levels. *true* and *false* examples are distributed evenly over the data. The pairs are not annotated for confidence score, as truth values are absolute and not questionable.

The data originate from various corpora of English newspaper text[3] but are manually adapted in order to create useful *T-H pairs* and are grammatically corrected if necessary.

2.4.1 The Seven Subsets

The seven subsets in the corpus all have different syntactic and semantic characteristics.

Information retrieval (IR): *H* is a prominent sentence from a news story and *T* a sentence from a retrieved web document. *H* describes a relationship between two things or people. Word overlap between *T* and *H* seems to be high for both *true* and *false* pairs.

Comparable Documents (CD): *T* and *H* are texts with lexical overlap from comparable news articles. Nouns appearing in *H* but not in *T* seem to be predictors of *false* pairs and *true* pairs seem to have a significantly higher word overlap.

[3]The following sources were used in the preparation of the data:

1. Document Understanding Conferences (DUC) 2004 Machine Translation evaluation data, from the National Institute of Standards and Technology (NIST),
2. TextMap Question Answering online demo, from the Information Sciences Institute (ISI),
3. Relation Recognition dataset, from University of Illinois at Urbana-Champaign,
4. DIRT paraphrase database (online demo), from the University of southern California,
5. Corpus of Sentence Alignment in monolingual comparable corpora, Columbia University.

Reading Comprehension(RC): Entailment pairs inspired by high school reading comprehension tests. T and H are often rather parallel in syntactic structure, but use synonyms. We do not expect word overlap to be a clue.

Question Answering (QA): Given a certain question, T is the text snippet in which the answer was found by a QA system, and H is the affirmative sentence wich would be a good output of a QA system. The entities and relations in H are often also present in T, but in other combinations. We expect word overlap to be a clue.

Information Extraction(IE): T is a text in which a certain semantic relation holds and H is a created natural language formulation of this relation. Sentences are often about business successions, wars and killings. Word overlap seems to be the same for *true* and *false* pairs.

Machine Translation (MT): T and H are an automatic translation and a gold standard human translation of the same text. T and H are of the same length and have a same syntactic structure. False friends are often present, as are abbreviations of names of organisations. Word overlap does not seem to be a good clue for entailment.

Paraphrase Aquisition (PP): T and H often have a different syntactic structure.

Note that the test corpus is annotated for the "task types" above, which makes it possible to differ the treatment of a *T-H pair* according to the task type. This would of course make the design less applicable in a real world setting; this annotation is artificial. We would, in that case, have to recognise the task type according to characteristics like the ones above.

2.4.2 Constructions and Inference levels

The syntactic and semantic phenomena that play a role in textual entailment are numerous. We have made an informal inventory by taking a random sample from the development data set, consisting of 4 *true* example pairs of each kind, that is 28 in total. We look for syntactic constructions in T and H that can be seen as connected by a chain of syntactic transformations and constructions (NPs or verbs) that are analogous or show another resemblance on the surface level. The phenomena that we come across in the sample corpus are the following:

(3) a. T: The extraditables today claimed responsibility for the murder of Antioquia police commander colonel Waldemar Franklin Quintero, which occurred this morning in Medellin.

b. H: police officer killed

(4) a. *T*: Hepburn's platinum, diamond and sapphire brooch had been estimated to fetch just $20,000, but sold for $120,000, six times its estimated price.

b. *H*: Hepburn's diamond and sapphire brooch fetched $120,000.

(5) a. *T*: C&D Technologies announced that it has closed the acquisition of Datel, Inc.

b. *H*: C&D Technologies acquired Datel Inc.

(6) a. *T*: "I guess you have to expect this in a growing community,"; said Mardelle Kean, who lives across the street from John Joseph Famalaro, charged in the death of Denise A. Huber, who was 23 when she disappeared in 1991.

b. *H*: John J. Famalaro is accused of having killed Denise A. Huber.

(7) a. *T*: Sultan Al-Shawi, a.k.a the Attorney, said during a funeral held for the victims, "They were all children of Iraq killed during the savage bombing."

b. *H*: The Attorney, said at the funeral, "They were all Iraqis killed during the brutal shelling."

Auxiliary ellipsis. Example 3 illustrates at least three different syntactic phenomena that we should consider if we would like our design to recognise the relation between *the murder of Antioquia police commander colonel Waldemar Franklin Quintero* in 3a and police officer killed in 3b. Auxilliary ellipsis is the first one. 3b is a typical headline: its auxiliary *be* is omitted. We should, however, be able to recognise that *killed* is a participle and not a past verb.

Nominalised verbs. *the murder of* in 3a is the nominalised form of *to murder*. Our design should be aware of this and turn this noun phrase into an active sentence in order to be able to recognise the correlation with 3b.

Passive/active. The last syntactic transformation that should be done on this sentence pair is turning the passive construction in 3b into an active one: *(. . .) killed police officer*.

Anaphora resolution. Anaphora resolution should be performed to interpret the meaning of anaphora like *it* in 5a.

Relative clauses. We have to make sure that noun phrases are connected to the relative clause that follows them, as in *John Joseph Famalaro, charged in the death of Denise A. Huber* in 6a.

Other patterns Of some patterns we know that they occur often, due to the composition of the corpus. We handle them as patterns, as in **??**: *Pribilof Islands in the Bering Sea* is transformed into *The Pribilof Islands are located in the Bering Sea*.

Spelling and punctuation. We should be able to recognise variation in the spelling or punctuation of proper names and other noun phrases, like in 5: *Datel, Inc.* versus *Datel Inc.*

Synonymous and analogous NPs. Synonyms are rarely real synonyms, and we consider them as being analogous constructions. An example is *bombing* versus *shelling* in 7.

In the next chapter, present our approach to this task. Our goal is to design a fairly accurate system, rather than handling each of the linguistic phenomena that play a role in textual entailment at any price.

Chapter 3

The Design

3.1 Introduction

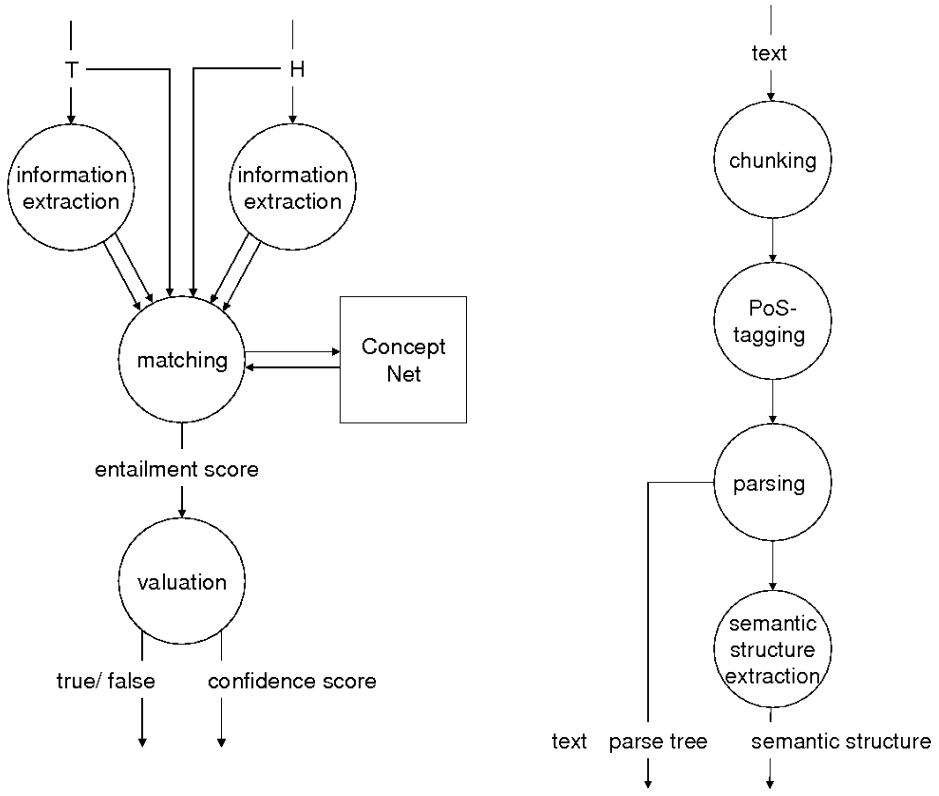
This chapter describes the design of our approach to the RTE task. In Section 3.2 we first present the overall picture of the design; an interpretation procedure containing existing and new tools and methods. Secondly, in Section 3.3 we give a detailed description of some of the modules in this procedure.

3.2 Overview and Data Flow

We are interested in how accurate a short English newspaper text or a sentence can be interpreted by extracting the assertions made in them, regarding only semantic structure, sentence modifiers and negations and using a conceptual network for recognising word-to-word conceptual relations. It is interesting to look how far we can get if we do this extraction thoroughly. Therefore, core of our interpretation mechanism is a predicate-argument extraction based on a detailed parse tree. We choose to represent the meaning of a text as a conjunction of the interpretations of the different sentences and subsentences in it and to represent the meaning of a sentence as a conjunction of assertions. These assertions will have to contain a predicate, one or more primary arguments and sentence modifiers, and negations and reflect the basic statement made in the sentence. We consider the texts to be conjunctions because we expect them to represent lists of facts rather than of possibilities or choices.

Figure 3.1(a) gives an overview of the data flow in our design. Circles represent processes and arrows data flow, name tags of arrows describe the flowing data. Squares are external entities, communicating with the system but not really being part of it.

Input is a *T-H pair*, output are a truth value and a confidence score. T and H each go through the interpretation mechanism in parallel, after which the resulting semantic



(a) Overview

(b) Data flow diagram for our Information Extraction procedure

Figure 3.1: Data flow diagram for our design.

structures are matched with each other using the notion of conceptual distance (querying the ConceptNet common sense database), resulting in a score indicating the level of entailment. Finally, a valuation algorithm assigns a truth value and a confidence score to the *T-H pair*, based on the entailment score.

The Information Extraction part of the design is given in Figure 3.1(b). First, the input text is chunked, PoS-tagged and parsed. After this, we extract a semantic structure.

Our design has been implemented and the resulting system has been trained over the development data. A final evaluation set which is provided with the RTE Challenge has been used for evaluation of the system. We refer to Chapter 6 for a detailed evaluation.

3.3 Interpretation Procedure

The interpretation procedure contains the following processes which are worked through in a linear order; chunking, tagging, parsing, matching and validation. The different stages of the interpretation procedure are illustrated in Figure 3.2.

3.3.1 Pre-processing

As *T-H pairs* are xml-coded, the first step in pre-processing is extracting T and H as raw text. Both T and H are also tagged for task-type, this information is saved and used later in the procedure.

Chunking

T of H are now strings of characters and punctuation marks, and have to be chunked into words to be input to the PoS-tagger later. These strings could contain more than one sentence, and we have to keep these together, recognising dots as sentence-ending marks. This is not trivial because of the existence of dots after abbreviations, capital marks in names, and sentence-ending dots and sentence-preceding capitals sometimes being absent.

We use our own chunking algorithm, which replaces abbreviations and contractions by their full form, interprets punctuation to see where sentences end and where not and replaces characters that would give problems when using our PoS-tagger or parser later on.

1. Replace all constructions that are in a list of abbreviations and contractions by their full form. *U.S.* for example is replaced with *United States*.
2. Replace all brackets by comma's, question- and exclamation marks by dots and dashes or sequences of lines by a comma. Remove quotation marks.
3. Turn spaces into line breaks when preceded by a sentence-ending character.
4. Leave spaces in when the preceding character does not end the sentence, e.g. in case of numbers like 1.50.

Tagging

After chunking a text, we use Thorsten Brants' TnT tagger [Bra00] to annotate each word and punctuation mark in the text with Part-of-Speech information. We refer to Appendix A for the used tag set [MSM94]. TnT[1] (Trigrams'n'Tags) is a fast tagger, trainable on corpora in almost any language and with almost every possible tag set. It has been pretrained on the German NEGRA and the English Susanne corpora and the (Wall Street Journal (WSJ) part of the) Penn Treebank. We use the WSJ part because of its performance (accuracy of about 96,7% on the WSJ) and the fact that its tag set is the same as the one our parser has been trained on.

Parsing and Semantic Structure Extraction

After tagging, we run Julia Hockenmaier's wide coverage StatCCG parser [Hoc03] over TnT's output.

[1] http://www.coli.uni-saarland.de/ thorsten/tnt/

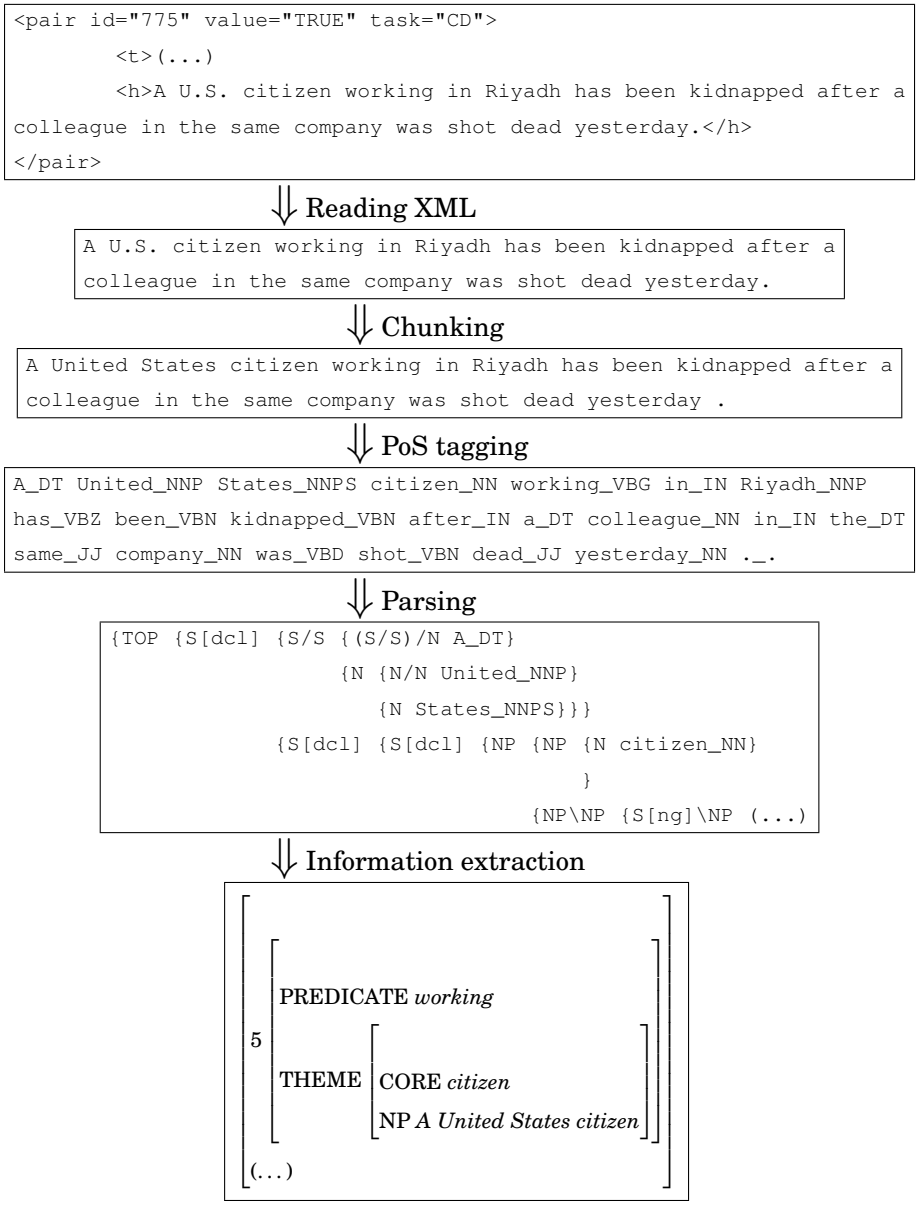


Figure 3.2: Different stages in the processing of a *T-H pair*.
From top to bottom: xml-coded *T-H pair* and for *H* the text version, PoS-tagged version, part of the parse tree, part of the extracted semantic structure respectively.

In Chapter 4, StatCCG and its output are discussed in detail, as is our procedure for the extraction of semantic information.

3.3.2 Matching

The resulting semantic structures are matched to one another in order to calculate their agreement, using a common sense database for the calculation of the conceptual distance between words. See Chapter 5 for a discussion of this procedure.

Output of the matching procedure is a number called the *entailment score*, which we use for determining a truth value and a confidence score.

3.3.3 Validation

Truth value of a *T-H pair* is *true* if its entailment score is on or above the *entailment threshold* and *false* if not. The entailment threshold is calculated by optimising the model as will be explained in chapter 6.

The confidence score of the pair is defined as the relative distance of the entailment score to the entailment threshold, and it expresses the confidence of the algorithm in its judgement. It is calculated directly out of the entailment score by simply taking the distance between entailment score and entailment threshold relative to the difference between the maximum entailment score (for pairs judged as *true*) or minimal entailment score (for pairs judged as *false*) and the entailment threshold. Maximum and minimum entailment scores are also determined during optimisation of the model.

$$\textit{confidence score} = \begin{cases} \frac{|(\textit{threshold - entailment score})|}{|(\textit{threshold - highest entailment score}|} & if\ \textit{entailment score} > \textit{threshold} \\ \\ \frac{|(\textit{threshold - entailment score})|}{|(\textit{threshold - lowest entailment score}|} & if\ \textit{entailment score} < \textit{threshold} \end{cases}$$

Chapter 4

Information Extraction

4.1 Introduction

Verbs, arguments and their semantic roles form an important part of the meaning of a sentence, and as Gildea & Palmer [GP01] concluded, the output of wide-coverage parsing is, although computationally expensive, more effective for the purpose of identifying semantic roles than flatter structures. In this chapter we present an algorithm for extracting semantic information from a text; head verbs, first order arguments and their semantic roles, sentence modifiers and negative elements. Our method only uses the head-satellite structure and parse trees obtained from parsing with a statistical CCG parser.

The structure of this chapter is as follows. First, in paragraph 4.2, we will introduce the parser that we will be using for our syntactic analysis and explain why this is a good choice. We also explain what the parser output looks like. The information extraction algorithm is then described in 4.3.

4.2 A Statistical CCG Parser

StatCCG[1] [Hoc03] is a recent wide-coverage parser, written by Julia Hockenmaier as part of her dissertation project, and has been pre-trained on CCGBank, a translation of the Wall Street Journal (WSJ) into a CCG tree bank. This Treebank has been extracted from the WSJ using a statistical model. The parser has an accuracy of about 83% on word-word relations (labelled recovery) and 91% (unlabelled recovery) in unseen text, which is about as high as most of the modern parsers used at the moment. Input for the parser has to be PoS-tagged to WSJ-standards. Output is a parse tree and what Hockenmaier calls a *predicate-argument structure* for every sentence. We use the term *head-satellite structure* for this.

[1] http://www.ircs.upenn.edu/ juliahr/Parser/

Combinatorial CG [Ste00] is an extension of Bar-Hillel's Categorial Grammar theory [BH53]. CCG is a lexicalised surface-structure grammar formalism based on the Categorial Calculus and Combinatory Logic, in which every syntactic structure is associated with a fully parallel semantic structure, representing its interpretation. One of the main advantages of CCG, as opposed to alternatives like phrase structure grammar, is its simple and semantically transparent treatment of long-range dependencies, made possible by control, raising, coordination and extraction. StatCCG's excellent performance and the fact that it has been trained on texts of a similar kind and from similar sources as the ones in our data set are the reasons for choosing StatCCG for out project.

Hockenmaier [Hoc03] gives a confusion matrix containing the most common lexical categories and the categories StatCCG proposes instead. She also gives an overview on its performance on the various predicate-argument relations. It shows that non-local dependencies are harder than local dependencies and the parser has some difficulties on distinguishing between PP complements and adjuncts. This should not be a great problem when extracting verbs and their NP arguments.

4.2.1 StatCCG Output

StatCCG returns, for every sentence, the most plausible parse tree and a head-satellite structure which includes all words in the sentence and is derived directly from the parse tree. For Example 8a, a fragment of the parse tree is given in Figure 4.2.

(8) a. *T*: An Iraqi official reported today, Saturday, that 68 Iraqi civilians were killed today as a result of the American and British bombing on Iraq and that their funerals were held today in Baghdad.

b. *H*: Reports from an Iraqi official today, Saturday, said that the 68 civilians killed as a result of the Anglo-American bombing of Iraq, were buried today in Baghdad.

In CCG, the syntactic behaviour of a word is defined in the lexicon by its lexical category, which is either an atomic category like `N` or a complex functor category built out of atomic categories and backward and forward slashes, as in `NP\N`. Rules provide for the combination of words to phrases. 'killed' in Example 8 has category `S[pss]\NP`, indicating a passive sentence `S[pss]` lacking a noun phrase `NP` to the left side. If 'killed' would have a noun phrase directly to the left, this would be its satellite and they would be able to form a passive sentence together. Or take the preposition 'as'. Its lexical category is `((S\NP)\(S\NP))/NP`, containing three 'slots' that can matched with phrases in order to build a sentence: first an `NP` to the right, then an `S\NP` (a verb phrase) to the left and finally an `NP` to the left. As can be seen in Figure 4.2, the third slot from the left is filled with the `NP` 'a result of the American and British bombing', establishing a relation between

'as' and the nominal head of the NP 'result' of type 3. The second is filled with the passive VP 'killed', resulting in a relation of type 2. Figure 4.1 shows part of the StatCCG output for Example 4.1 and is a list of all word-word relations in this sentence. Every row represents the relation between a head and a satellite. The fifth and sixth column from the left represent satellite and head, the first and second column their place in the sentence, the third column the category of the head and the fourth column the index of the satellite. The relations between 'as' and 'result' and 'as' and 'killed' can be found on row 16 and 14 respectively.

4.2.2 Parser Output for our Development Data Set

After pre-processing the development data as described in Section 3.3.1, all sentences (about 1100) in our development set can be parsed by StatCCG. Nearly all are parsed as declarative sentences, but there are also some passive sentences (11, of which 8 unique ones due to multiple instances of exactly the same sentence), to-infinitives (3), b-sentences (bare infinitives, subjunctives or imperatives, 8 ones), 4 past participles in active mode (pt) and one adjectival sentence. However, this adjectival sentence contains a tagging error and should in fact be an imperative.

Not all sentences are parsed as an S. Of the 28 sentences in our development data not being parsed as sentences, 27 are parsed as NPs and one as a PP. Of the ones parsed as NPs, 17 (15 unique ones) contain a verb that is not recognised as such (12 (10 unique) times NNS, 3 times NN and one time NNP). The other 10 sentences (9 unique ones) are nouns, noun phrases or interjections that do not contain a verb at all (4 times), ambiguous sentences that contain a verb but for which the parser chooses to parse them as NPs (2 times) and elliptical sentences in which the only verb is deleted (4 times, 2 unique). In the PP-sentence, the head verb is PoS-tagged NNS and a 'dominant' proposition causes the sentence to be parsed as PP.

Our data very possibly also contains sentences that are parsed as real sentences, containing verbs and arguments, bur that nevertheless contains errors due to tagging and parsing errors. We have not yet been able to make a good investigation, but expect 1. some verbs not being recognised as verbs and 2. some verbs to have lost connection with their argument(s), particularly in case of non-local dependencies arising through (particularly object-) extraction.

```
<s> 34
2     0     NP[nb]/N                 1     official_NN    An_DT
2     1     N/N                      1     official_NN    Iraqi_JJ
2     3     (S[dcl]\NP)/S[em]        1     official_NN    reported_VBD
3     4     (S\NP)\(S\NP)            2     reported_VBD   today_NN
3     6     (S\NP)\(S\NP)            2     reported_VBD   Saturday_NNP
8     3     (S[dcl]\NP)/S[em]        2     that_IN        reported_VBD
11    9     N/N                      1     civilians_NNS  68_CD
11    10    N/N                      1     civilians_NNS  Iraqi_JJ
11    12    (S[dcl]\NP)/(S[pss]\NP)  1     civilians_NNS  were_VBD
11    13    S[pss]\NP                1     civilians_NNS  killed_VBN
12    8     S[em]/S[dcl]             1     were_VBD       that_IN
13    12    (S[dcl]\NP)/(S[pss]\NP)  2     killed_VBN     were_VBD
13    14    ((S\NP)\(S\NP))/NP       2     killed_VBN     as_IN
13    23    ((S\NP)\(S\NP))/NP       2     killed_VBN     on_IN
16    14    ((S\NP)\(S\NP))/NP       3     result_NN      as_IN
16    15    NP[nb]/N                 1     result_NN      a_DT
16    17    (NP\NP)/NP               1     result_NN      of_IN
22    17    (NP\NP)/NP               2     bombing_NN     of_IN
22    18    NP[nb]/N                 1     bombing_NN     the_DT
22    19    N/N                      1     bombing_NN     American_JJ
22    21    N/N                      1     bombing_NN     British_JJ
24    23    ((S\NP)\(S\NP))/NP       3     Iraq_NNP       on_IN
26    3     (S[dcl]\NP)/S[em]        2     that_IN        reported_VBD
28    27    NP[nb]/N                 1     funerals_NNS   their_PRP
28    29    (S[dcl]\NP)/(S[pss]\NP)  1     funerals_NNS   were_VBD
28    30    S[pss]\NP                1     funerals_NNS   held_VBN
29    26    S[em]/S[dcl]             1     were_VBD       that_IN
30    29    (S[dcl]\NP)/(S[pss]\NP)  2     held_VBN       were_VBD
30    31    (S\NP)\(S\NP)            2     held_VBN       today_NN
30    32    ((S\NP)\(S\NP))/NP       2     held_VBN       in_IN
33    32    ((S\NP)\(S\NP))/NP       3     Baghdad_NNP    in_IN
<\s>
```

Figure 4.1: Enumeration of all head-satellite pairs in StatCCG's output for Example 8a. The first row contains the number of words in the sentence. The columns represent, from left to right: word number of the satellite, word number of the head, complex category of the head, type of head-satellite relation, satellite plus PoS-tag, head plus PoS-tag.

```
{S[pss]\NP {S[pss]\NP killed_VBN}
           {(S\NP)\(S\NP) {((S\NP)\(S\NP))/NP as_IN}
                          {NP {NP {NP[nb]/N a_DT}
                                  {N result_NN}}
                              {NP\NP {(NP\NP)/NP of_IN}
                                     {NP {NP[nb]/N the_DT}
                                         {N {N/N {N/N American_JJ}
                                                 {N/N[conj] {conj and_CC}
                                                            {N/N
British_JJ}}}
                                            {N bombing_NN}}}}}}}
```

Figure 4.2: StatCCG parse tree for "killed as a result of the American and British bombing" in Example 8a.

4.3 Extraction of Semantic Structure from the Parser Output

The goal is to extract from a text all meaningful verbs and their first order arguments and their semantic roles, sentence modifiers, and information about whether predicates are negated.

Because of the deictic nature of the data (news paper texts often containing temporal expressions relative to the time of print, which is unknown here), tense can be neglected for the task of RTE. This also means that modals and auxiliaries will not have to be part of the semantic representation of a text. The sentences "John will see him", "John saw him", "John has seen him", "John sees him", "John likes to see him", would, for example, all get the same interpretation. We, however, allow deictic sentence modifiers like "Saturday", because it is not easy to distinguish them from non-deictic modifiers.

4.3.1 The Algorithm

Input to our extraction algorithm is a text (*T* or *H*) and its head-satellite structure(s) and parse tree(s) from the parser output. Output for every text is an Attribute-Value Matrix (AVM) representing its semantic structure. Figure 4.3 shows its general form, containing all of its possible attributes.

The semantic representation contains list of predicate structures, each containing the attribute PREDICATE which has a value a verb of the form $verb\_PoS$. A predicate structure also contains one optional AGENT, PATIENT and THEME each containing an NP argument of the verb (the whole NP as well as its head N), zero or more MODIFIER attributes with as value a sentence modifier phrase, and the optional attributes PASSIVE (with value *passive*), NEGATION (with value *negation*) and VERBCLUSTER, which occur

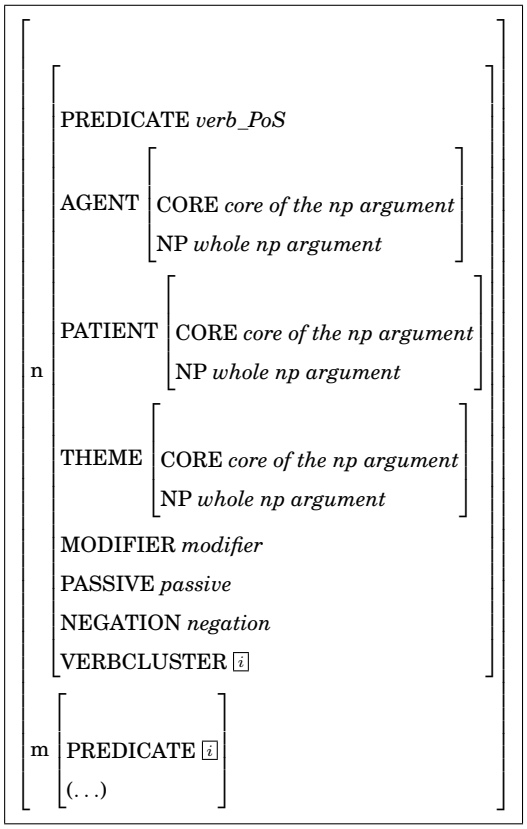


Figure 4.3: General form of a semantic representation AVM

only once or not at all in a predicate structure. The value of VERBCLUSTER is shared with the value of one of the PREDICATEs in another predicate structure. This is called *reentrancy*. Each predicate structure is identified by a unique number n, which is larger or equal to 0, indicating that the verb which forms the value of the PREDICATE attribute is the n^{th} word in the text.

Our semantic structure extraction algorithm is the following:

1. for every text, construct an AVM, containing:
2. for every verbal head in all of the head-satellite structures for this text, add a predicate structure containing:
3. ... its primary argument(s) and their semantic roles,
4. ... sentence modifiers, if present,
5. ... verb clusters, if present,
6. ... if the verb is negated or not
7. ... if the verbal head is passive or not
8. for every AVM, for every predicate structure,
9. if it contains a verb cluster, move its content to the predicate structure it is coindexed with and remove the predicate structure

As texts can contain more than one sentence, a text can correspond to more than one head-satellite structure. We store all of the verbs in the head-satellite structures that belong to a certain text in the same AVM, and identify the verbs by numbering the predicate structures by the place of the verb in the text, which is equal to the place of the word in the sentence (see the first row of the head-satellite structure), increased by the number of words in all of the preceding sentences in the text.

We only store verbal heads that have a noun phrase, personal pronoun or comparative adverb as satellites. This satellite is stored as core of an argument as the value for the CORE attribute. The full argument, however, is extracted from the parse tree of the sentence by looking for the largest noun phrase in which the CORE noun is present and the verbal head is not. The semantic role of the argument will be AGENT if its index in the head-satellite structure is 1, PATIENT if its index is 2 and THEME if the verb is passive. Passive verbs can be recognised by the *voice* feature $[pss]$ in their lexical category. If the satellite of a verbal head is a verb (but no modal verb) itself, we say that the verbal head and satellite form a verb cluster. This information is added to the AVM by coindexing the predicate structures of head verb and satellite verb.

Sentence modifiers are heads with category `(S\NP)\(S\NP)}`, or the smallest NP that a head with category `((S\NP)\(S\NP))/NP}` is a head of.

If the head of a head-satellite pair is a negating element (e.g. *n't, not, never*) and the satellite a verb, this negation is added to the predicate structure of the verb by adding "NEGATIVE *negative*".

At this point, we have build a semantic structure like the one in Figure 4.4 (for the text in Example 8a).

Now that we have collected all desired information, we remove all auxiliaries from the AVM by adding, for every predicate structure containing a VERBCLUSTER attribute, all information to the coindexed predicate structure and removing the predicate structure with the verb cluster:

$$\left[\begin{array}{l} \text{n}\ \left[\begin{array}{l}\text{PREDICATE } \textit{verb1}\\ \textit{list of attributes } l\\ \text{VERBCLUSTER } \boxed{i}\end{array}\right] \\ \text{m}\ \left[\begin{array}{l}\text{PREDICATE } \boxed{i}\ \textit{verb2}\\ \textit{list of attributes } l2\end{array}\right] \end{array}\right] \longrightarrow \left[\text{m}\ \left[\begin{array}{l}\text{PREDICATE } \textit{verb2}\\ \textit{list of attributes } l + l2\end{array}\right]\right]$$

For example 8a, this results in the predicate structures
12={PREDICATE=were,THEME=68 Iraqi civilians,VERBCLUSTER=i} and

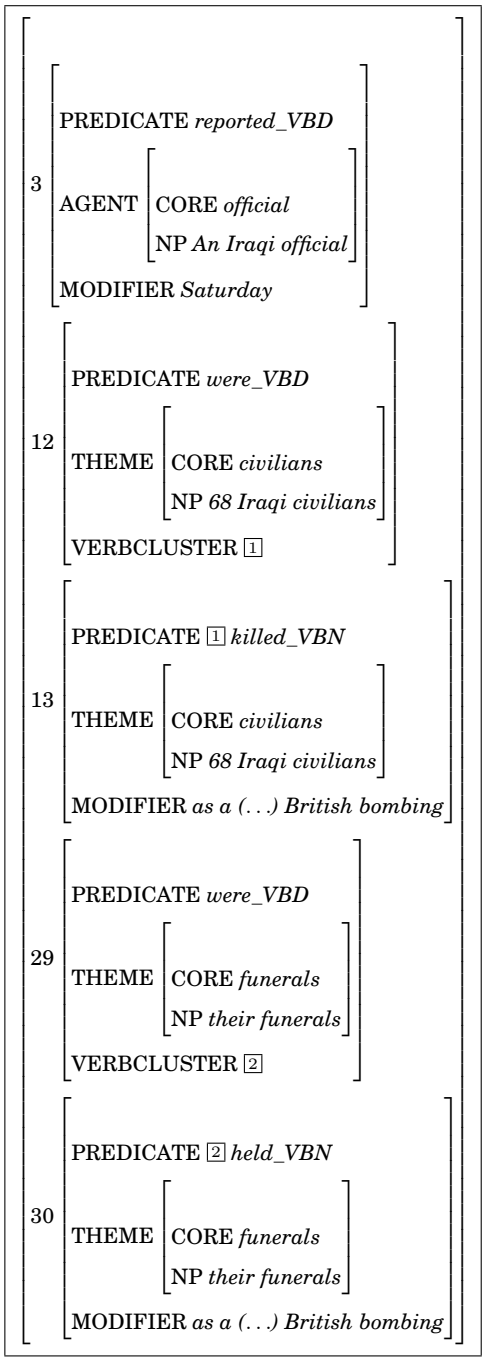


Figure 4.4: Extracted semantic structure after step 1.

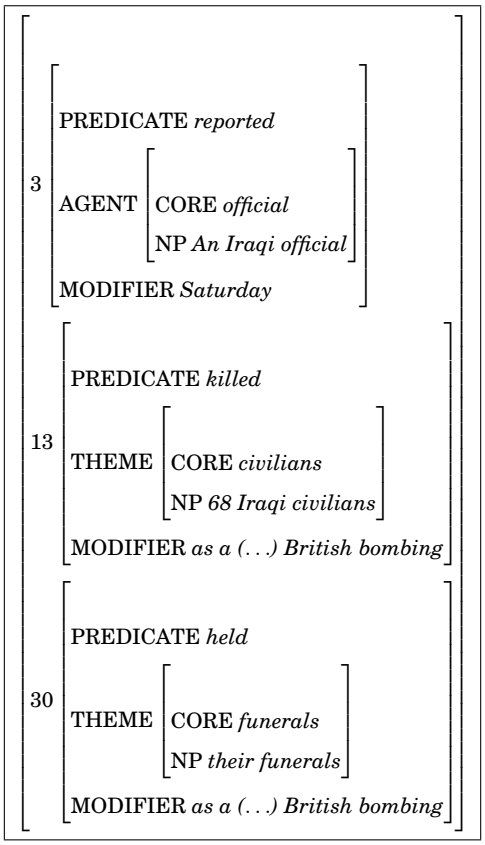


Figure 4.5: Extracted semantic structure after step 2.

13={PREDICATE=killed$_i$,THEME=68 Iraqi civilians,MODIFIER=as a result of the American and British bombing} being replaced by predicate structure 13 and the predicate structures 29 and 30 by predicate structure 30. See Figure 4.5.

Furthermore, if a predicate structure contains a PATIENT and no AGENT, we change the semantic role of this argument to AGENT, because this is probably due to a parsing error.

For Example 8a, we have now come to the semantic structure in Figure 4.5.

Identifying Semantic Roles Using PropBank

Gildea and Hockenmaier [GH03] also propose a system for automatically identifying semantic roles based on the StatCCG head-satellite structure. They perform a mapping between PropBank semantic roles and CCG head-satellite relations in order to be able to use CCGBank as input to their semantic role-labelling system. PropBank [KP02] is a human-annotated corpus of semantic verb-argument relations, similar to FrameNet but without frames. The system is based on a similar system by Gildea & Palmer [GP01], which uses the Collins parser instead [Col99]. The system does not only recognise primary semantic roles, but also the secondary roles *Location, Time, Manner, Direction, Cause, Discourse, Extent, Purpose, Negation, Modal* and *Adverbial*. Semantic roles are recovered by considering the index of the slot that the satellite fills in the lexical category of the head, like we did, or by comparing the parse tree paths that are traversed when going from head to satellite. Mapping CCG relations to PropBank relations is not trivial, mainly because of differences in the definition of the head of a constituent.

Gilea & Hockenmaier's system outperforms a system using a traditional parser for the primary arguments, partly because StatCCG recovers long-range dependencies, which are particularly important for primary arguments. Primary semantic roles represent 75% of the total labelled roles in PropBank. The system, however, performs lower on adjuncts.

Chapter 5

Common Sense Reasoning

5.1 Introduction

In this chapter, we focus on the alignment of the semantic structures that we extracted in Chapter 4 and on how we use the common sense database ConceptNet for calculating the semantic distance between words.

5.2 Common Sense Reasoning

One of the dreams of researchers in Artificial Intelligence is to be able to provide their agents with common sense. Common sense is commonly distinguished from expert knowledge[1]; its domain is shallow but broad where expert knowledge is deep but narrow, its reasoning fail-soft opposed to the often unidirectional reasoning of expert models, and its knowledge is understandable not only by experts. It is the knowledge of everyday life, rarely written down explicitly so not or only difficult recoverable, and therefore assumed to be present as a condition for human- or human-like interpretation. Common sense is often about the temporal, spatial and causal context of actions or objects ("if someone sleeps, he usually does not stand upright" and "things fall down and not up") and would enable agents to perform much more intelligent reasoning about its context and the domain-specific knowledge it may already have. In recent years, researchers (for example Minsky, one of the founders of AI, and Lenat) have thought of common sense to be vital for Artificial Intelligence and have began collecting large amounts of common sense knowledge in databases.

Semantic knowledge bases also become more and more important in language techno-

[1]Another type of knowledge commonly distinguished is *novice knowledge*; its domain is shallow and narrow but restricted, its reasoning restricted and its contents understandable only for experts in the domain.

logy. The most widely used one in language technology is WordNet[2], a semantic ontology containing words, primarily nouns, verbs and adjectives, organised into discrete senses, and linked by a small set of semantic relations such as the synonym relation and is-a hierarchical relations. Its most recent version 2.0 contains roughly 200 000 word senses (a sense is a distinct meaning that a word can assume). One of the reasons for its success and wide adoption is its ease of use. As a simple semantic network with words at the nodes, it can be readily applied to any textual input. In *RTE1*, 10 of the 28 approaches that were send in used WordNet, 3 used other world knowledge like collocation information from the internet or small, hand-written knowledge bases, inspired by the development data.

An example of a similar project is FrameNet[3], in which concepts or words are grouped together in situation types having their own specific semantic behaviour, the so-called *semantic frames*. An interesting phenomenon however, is the development of more associative conceptual networks like the common sense database and reasoning system Cyc[4] and also Mindpixel[5] and ThoughtTreasure[6]. Many of these projects have integrated WordNet's ontology.

5.2.1 ConceptNet

ConceptNet[7], the system we used as a source of common sense in our design, developed at MIT Media Lab, is a conceptual network inspired by both WordNet and Cyc and contains about 1.6 million assertions. The ConceptNet database (also known as OMCS-net) was automatically generated out of the Open Mind Common Sense Repository[8], a collection of nearly 700.000 English sentences of common sense facts brought together by over 14000 volunteers in a fill-in-the-gap fashion (e.g. 'A spoon is used for...').

ConceptNet is a conceptual network; its nodes are natural language expressions which are connected to one another by a total of 20 different binary relation types. Nodes can be single words or small word groups; noun phrases, verbs, prepositional phrases and adjectival phrases, spellchecked and stripped of any determiners, modals, and other semantically peripheral features. Figure 5.1 shows how one of the concepts from our example is embedded in ConceptNet. It shows some of the paths that connect the different concepts. We see that *funeral* and *bury*, the concepts that have to be linked in order to establish a relationship are connected in several ways. The semantics of the relations in ConceptNet is informal; the syntax and semantics of their arguments are not restrained in any way. This

[2] http://wordnet.princeton.edu/

[3] http://framenet.icsi.berkeley.edu/

[4] http://www.cyc.com

[5] http://www.mindpixel.com

[6] http://www.signiform.com/tt/htm/tt.htm

[7] http://www.conceptnet.org/

[8] http://openmind.media.mit.edu/

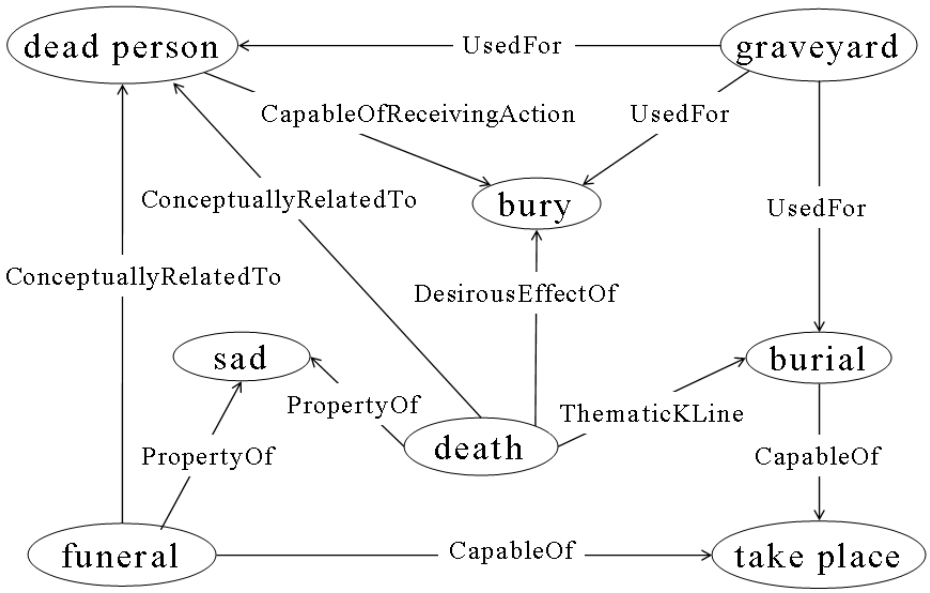


Figure 5.1: Embedding of the concepts *funeral* and *bury* in ConceptNet. Only part of the paths are given.

makes it possible to calculate the distance along one or more path types between concepts of different kinds. The natural language nodes do not have a real meaning; its meaning is expressed by the relations that exist between the node and other nodes in the network.

See [LS04a] for a discussion of ConceptNet's inferencial strength and [LLSB04] for some examples of the use of ConceptNet in actual applications.

5.3 ConceptNet versus Wordnet

ConceptNet is useful for recognising the ontological, morphological and world knowledge inferences on the lexical (and small word group) level. But why choose to use ConceptNet and not WordNet, as is done several times in *RTE1* and worldwide in natural language processing? We have some important reasons to do so.

ConceptNet is structurally very much like WordNet, and can also be used in a similar way by looking for semantic chains between concepts by network traversal, but there are a few important differences. The first one is that it is richer. ConceptNet contains, like WordNet, natural language nodes, connected with edges that represent relations, but where WordNet has three such relations that make it a formal taxonomy (is-a, part-of and synonymy), ConceptNet has a much richer set of 20 relations. Moreover, it contains not only single- but also multi-word expressions including small predicates (like "eating

food"). However, it must be said that ConceptNet does not distinguish between senses yet, as WordNet does, because of it being extracted from a corpus that is not sense-tagged. Work is done at the moment to fill this gap. The second reason is that WordNet is a strict, hierarchically ontological network where ConceptNet is much more about association. Its knowledge is less hierarchic and more defeasible. This is the case partly because of the democratic and somewhat anarchistic acquisition, where WordNet was hand-crafted by a group of experts, partly because ConceptNet is dedicated to contextual reasoning. Of the 1.6 million assertions in it, 1.25 million are generic conceptual connections called k-lines , used for analogy-reasoning, for example about unknown concepts.

Finally, we think it is interesting to put ConceptNet to the test in a concrete application, as common sense knowledge bases are not easily evaluated. See [LS04b] for some work on evaluating ConceptNet and the repository it originated from.

5.4 Calculating Semantic Distance

The reasoning algorithm takes an attribute-value matrix like in Figure 4.5, here given again as Figure 5.2 as input. It is the Text part of pair 355 in our development data. Figure 5.3 is its Hypothesis part. As we can see, they both consist of three predicate-argument structures that represent assertions. The semantic representation of a text here usually contains one to five assertions, Texts being normally longer than Hypotheses. Output of the algorithm will be the entailment score for every pair (in a range from 0 to 1), representing the probability of the entailment relation being present.

Seeing T and H as sets of assertions, entailment exists if it is likely that the sum of all assertions in H is true given T. Have a look at example 9. Say, the probability that the sun is shining if the weather is beautiful is 80% and the probability that it will be hot if the weather is beautiful is 50%. The probability of H given T cannot simply be $80\% * 50\% = 40\%$, because we do not know if "the sun shines" and "it's hot" are dependent. Would T and H be the other way around, this problem would exist also. Every assertion in T and in H would have to be checked for dependencies.

(9) a. T: The weather's beautiful

b. H: The sun shines and it's hot.

Therefore we will assume that all assertions are independent. We will try to prove entailment by proving for each assertion in H that it is entailed by the assertion in T most probable of entailing it.

There is something very interesting about our development data. Due to the symmetry that the developers have brought into it, long sentences have an equally large probability of being tagged as *true* as short ones. If we would calculate the probability of a relation by

simply multiplying the probabilities of the single assertions, we would expect that longer sentences would have a much smaller average probability and therefore be judged as *true* less. Because of this, we choose to take the average probability instead of the product of probabilities.

A possible way of calculating conceptual distance between two texts would be to calculate the ConceptNet conceptual distance between every pair of assertions and adding up the values for the closest pairs. ConceptNet, however, is an expensive tool in that it is very time-extensive. We therefore choose to use word- and structure matching first to align assertions and use ConceptNet only for these aligned sentences.

First, we will try to align the assertions in T and H.

5.4.1 Assertion Alignment

We align assertions by going through the list of assertions in H and counting the number of matching predicates and arguments and modifiers with each of the assertions in T. Negations are ignored in this stage. Predicates, arguments or modifiers match if their base form is exactly the same. The base form of a word (singular for nouns and base form verbs) we get from querying the `lemmatise` method in ConceptNet. For arguments, we take the base form of their CORE (opposed to the elaborate version).

The assertion in H with the best match with one of the assertions in T gets the first pick; it is aligned with the assertion that it matches best with. The second best assertion in H gets the second pick, etcetera. If two assertions match the same assertion in T equally, the first one takes precedence.

Let us have a look at Figure 5.2 and 5.3. Assertion 14 in H is aligned to assertion 13 in T. Assertions 9 and 26 are not matched at one of the assertions in T at all, so they are aligned to the first and the second still unmatched assertions in T respectively; 3 and 30.

5.4.2 Conceptual Distance

Now that we have aligned the assertions in T and H, we calculate the conceptual distance[9] between the aligned assertions. We do this by querying the ConceptNet method `get_context` for the two predicates and the pairs of arguments that are present.

`get_context` walks the conceptual network like a spider: 'Technically speaking, the contextual neighbourhood around a node is found by performing spreading activation radiating outward from that source node. The relatedness of any particular node is not simply a function of its link distance from the source, but also considers the number and strengths of all paths which connect the two nodes."[LS04b]. `get_context` contains a vector of relation types to weights varying from 0.0 to 1.0, which can be biased towards, for example,

[9]Conceptual distance is, in fact, conceptual *nearness*; the larger the value, the more close two concepts are to one another

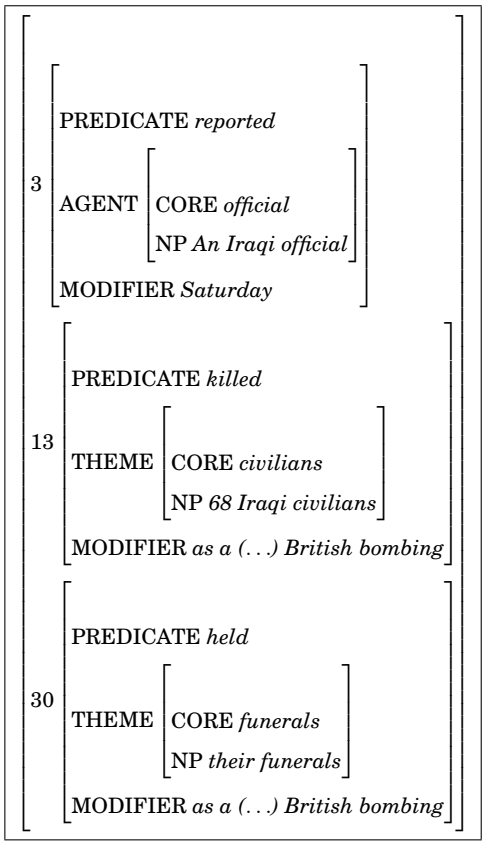


Figure 5.2: Extracted semantic structure for Example 8a.

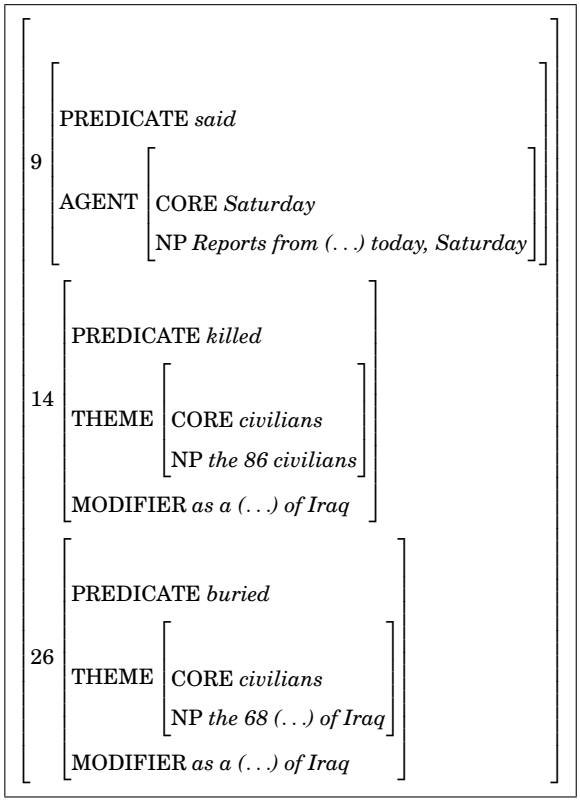


Figure 5.3: Extracted semantic structure for Example 8b.

```
[[report, 0.401], [hold, 0.073],[have examination, 0.063], [statement,
0.057], [spring binder, 0.053], [pass class, 0.041], [read, 0.046],
[report crime, 0.050], [witness, 0.058], [take examination, 0.055],
[pass, 0.042], [victim, 0.044], [police, 0.036],
```

Figure 5.4: Example of a context list.
Fragment of (a possible) context of the concept *report*. We rounded the values to three decimals here for clarity.

spatial or causal context.

We walk the network up to a maximum of 300 nodes, and until we have at most 300 nodes in the output list. Figure 5.4 shows a possible context for the verb "report". Conceptual distance between "report" and "statement" is approximately 0.057.

For each predicate or argument in an assertion (in base form) in H, we calculate the context, we normalise the values in the list, which do not have much external meaning, to the range of 1 to 0.01 and search for the corresponding predicate or argument in the list. If it is not found, the value 0.01 is assigned. Then, for every assertion, all values are summed up. If, for two aligned assertions, one is negated and one is not, this causes a penalty of one time the conceptual distance of the negated concepts; a larger penalty for the negation of a close concept pair and only a small one for concepts that do not show much resemblance.

Another conceptual similarity measure in ConceptNet is `get_analogous_concepts`, which is in fact a conceptual *relatedness* measure (*car* is related but not similar to *gasoline*). This measure did not perform well on a series of word-word pairs from the training set.

Chapter 6

Evaluation

6.1 Introduction

In this chapter, we present the results of the evaluation we performed on the final model. We first introduce accuracy and Confidence Weighted Score as measures for evaluating a model in Section 6.2.

Section 6.3 gives the results of some tests that we performed on the model during development and discusses the consequences of these results for the final design.

In Section 6.4, the final model is then evaluated against the evaluation data set.

Finally, in Section 6.5, we introduce some of the other approaches to the RTE task and compare their design and performance with ours.

6.2 Evaluation Methods

One of the hypotheses of this project is that we can use our development data set for training our model, in order to improve its performance on the test set. In order to test this hypothesis, we train the model by optimising the threshold that distinguishes between *true* and *false*. In the RTE challenge, this is already anticipated for as the development set come divided into two sets of approximately the same size, named *dev* and *dev2*, both containing 50% pairs annotated *true* and 50% pairs annotated *false*. We use *dev* as training set and *dev2* as test set.

Training is done as follows. See Figure 6.1 for an imaginary situation. It represents a set of T-H pairs that have been assigned entailment scores by our interpretation algorithm. + and - represent the real truth values *true* and *false* that the pairs are tagged for. The threshold is a line that categorises the pairs into a *true* and *false* set. The further away an entailment score is from the threshold, the larger the confidence score is. We repeate the definition of confidence score:

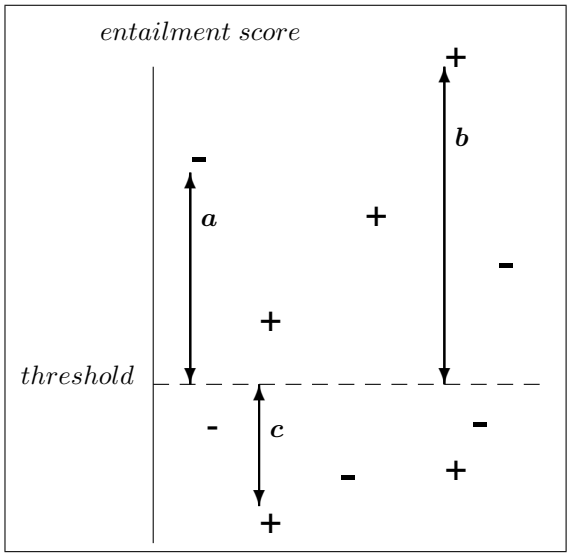


Figure 6.1: Entailment threshold, categorising an imaginary set of T-H pairs. + is a pair which is in fact (annotated as) *true*, - a pair which is in fact *false*, a = difference between an entailment score and the threshold, b = difference between the highest entailment score in the set and the threshold, c = difference between the lowest entailment score in the set and the threshold. The horizontal dimension has no meaning here.

$$\textit{confidence score} = \begin{cases} \frac{|(\textit{threshold - entailment score})|}{|(\textit{threshold - highest entailment score}|} & if\ \textit{entailment score} > \textit{threshold} \\ \\ \frac{|(\textit{threshold - entailment score})|}{|(\textit{threshold - lowest entailment score}|} & if\ \textit{entailment score} < \textit{threshold} \end{cases}$$

Maximum and minimum entailment scores are determined during optimisation of the model and cause the confidence score to be between 0 and 1.

We optimise the threshold by calculating the accuracy and Confidence Weighted Score (CWS) for a series of threshold values (in fact, the set containing the entailment scores of all pairs in the training set) and calculating the threshold that gives the highest CWS. This optimal threshold is used later for calculating confidence scores and CWS for the pairs in the test set. Figure 6.8 shows a training set with an optimal threshold of around 0.55, Figure 6.3 shows the resulting entailment score distribution when testing on this threshold.

The Confidence-Weighted Score ranges between 0 (no correct judgments at all) and 1 (perfect classification), and rewards the systems' ability to assign a higher confidence score to the correct judgments than to the wrong ones:

$$\textit{Confidence Weighted Score} = \frac{1}{n} * \sum_{i=1}^{n} \frac{\#-correct-up-to-pair-i}{i}$$

where n is the number of the pairs in the test set, and i ranges over the pairs.

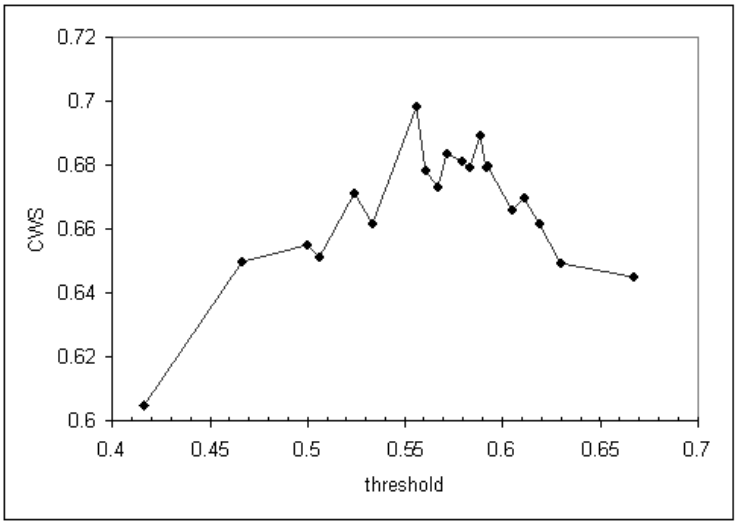


Figure 6.2: During training, the threshold is varied and CWS is calculated for each to threshold in order to find the optimal value. Optimal threshold in this case is 0.555.

As a second performance measure we calculate accuracy, which is the percentage of rightly judged pairs.

6.3 Development

We propose two more shallow algorithms, called *plain word overlap* and *lemmatised word overlap*. For plain word overlap, we glue the sentences in a text together, remove all punctuation, replace capitals by lower case characters and calculate the number of words in the resulting Hypothesis that are also in the Text, divided by the number of words in the Hypothesis. The lemmatised word overlap algorithm is different in that it uses the PoS-tagged texts instead of the raw texts, that we additionally remove determiners and modal verbs and that we lemmatise the words in the Hypothesis and the Text using the ConceptNet lemmatise function before we match them.

6.3.1 Performance

Table 6.2, 6.3 and 6.4 show the performance on the development test set for each of these three algorithms. They also show performance for each of the seven different task types separately. Table 6.1 shows the differences in composition between the training set and the test set. Note that the task types are not divided evenly over the two sets. *True* and *false* pairs are, although the amount of true pairs in the different subsets of the training set can sometimes be one more or less than the amount of false pairs, causing baseline to be slightly higher or less than 50%. Baseline for the total training set is also slightly less

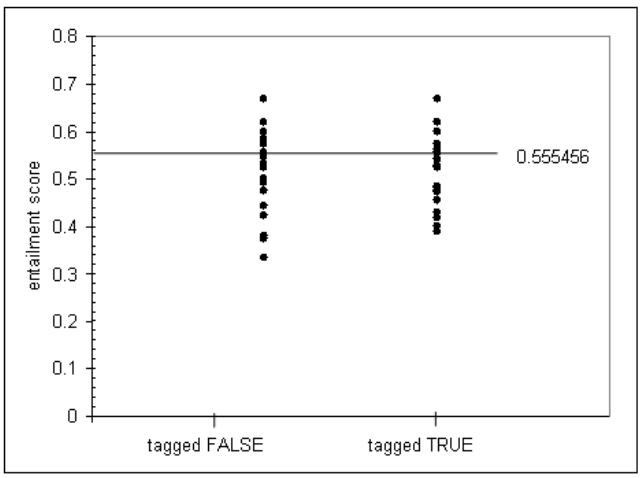


Figure 6.3: Example entailment score distribution for optimal threshold 0.555456

than 50%.

When we have a look at the Tables 6.2, 6.3, and 6.4, we see that accuracy and CWS are rather parallel in all cases. However, performance is very different for each of the task types. On CD, the model performs very well (accuracy of 79.2%, 83.3% and 62.5% respectively), as on MT (accuracy of 67.4%, 69.6% and 50.0% respectively). On the other task types, performance is around baseline or worse. On the test set as a whole, performance is only slightly above baseline (accuracy of 51.1%, 53.9% and 50.0% respectively). As to the performance for the three algorithms, lemmatised word overlap performs best, followed by plain word overlap and conceptual distance, which seems to perform randomly.

Striking are the very high Confidence Weighted Score for the CD subset for the first two algorithms. Because we expect threshold/CWS curves to be concave, optimal threshold values of around 0.3 to 0.7 are within expectations. It is remarkable that some of the threshold values are near to 1, as in Table 6.2 and 6.3 or to 0, as in Table 6.4. The percentage of pairs judged as *true* (relative to the total number of pairs in the set) shows that these thresholds are no good threshold for the test set. Accuracy and CWS are reasonably high, but only due to the fact that nearly all of the pairs are judged as *false*. Table 6.4 shows that conceptual distance is, in this form, no good measure for textual entailment. Apart from the subsets CD (relatively good result), IE (the algorithm seems to work here as a counter-argument for the entailment relation) and PP (66.7% judged as *true* but with accuracy 50%), the optimal threshold is the one judging 100% of the pairs as *true*. Which is mere guessing.

The differences in performance of the three algorithms on the seven subsets could be reflected in the final model as a weight distribution for the entailment scores of the pairs

	Training set		Test set	
	Occurrences	Baseline	Occurrence	Baseline
CD	50	52.0%	48	50.0%
IE	20	50.0%	50	50.0%
IR	50	50.0%	20	50.0%
MT	8	50.0%	46	50.0%
PP	58	46.6%	24	50.0%
QA	50	50.0%	40	50.0%
RC	51	51.0%	52	50.0%
all	287	49.8%	280	50.0%

Table 6.1: Amount of T-H pairs in and baseline for development training- and test set.

	threshold	accuracy	judged TRUE	CWS
CD	0.611011	38 (79.2%)	16 (33.3%)	0.933
IE	0.9999	24 (48.0%)	4 (8.0%)	0.420
IR	0.9999	9 (45.0%)	5 (25%)	0.363
MT	0.681718	31 (67.4%)	16 (34.8%)	0.725
PP	0.714186	12 (50.0%)	20 (83.3%)	0.502
QA	0.636264	11 (27.5%)	27 (67.5%)	0.217
RC	0.9999	21 (40.4%)	7 (13.5%)	0.317
all	0.666567	143 (51.1%)	167 (59.6%)	0.525

Table 6.2: Accuracy and CWS for plain word overlap algorithm when training and testing on development set. Accuracy is expressed as number of correct judgements and percentage of correct judgements in brackets.

	threshold	accuracy	judged TRUE	CWS
CD	0.4999	40 (83.3%)	22 (45.8%)	0.928
IE	0.2499	24 (48%)	3 (6.0%)	0.385
IR	0.9999	11 (55.0%)	5 (25.0%)	0.418
MT	0.7499	32 (69.6%)	17 (37.0%)	0.696
PP	0.9999	13 (54.2%)	3 (12.5%)	0.529
QA	0.666567	16 (40.0%)	32 (80.0%)	0.266
RC	0.3749	25 (48.1%)	49 (94.2%)	0.464
all	0.666567	151 (53.9%)	183 (65.4%)	0.563

Table 6.3: Accuracy and CWS for lemmatised word overlap algorithm when training and testing on development set. Accuracy is expressed as number of correct judgements and percentage of correct judgements in brackets.

	threshold	accuracy	judged TRUE	CWS
CD	0.083233	30 (62.5%)	20 (41.7%)	0.656
IE	0.166567	17 (34.0%)	18 (36.0%)	0.375
IR	-0.0001	10 (50.0%)	20 (100%)	0.397
MT	-0.0001	23 (50.0%)	46 (100%)	0.544
PP	0.0624	12 (50.0%)	16 (66.7%)	0.329
QA	-0.0001	20 (50.0%)	40 (100%)	0.412
RC	-0.0001	26 (50.0%)	52 (100%)	0.477
all	-0.0001	140 (50.0%)	280 (100%)	0.496

Table 6.4: Accuracy and CWS for conceptual distance algorithm when training and testing on development set. Accuracy is expressed as number of correct judgements and percentage of correct judgements in brackets.

	plain word overlap			lemmatised word overlap			conceptual distance		
	accuracy	judged TRUE	CWS	accuracy	judged TRUE	CWS	accuracy	judged TRUE	CWS
CD	38 (79.2%)	20 (41.7%)	0.941	41 (85.4%)	19 (39.6%)	0.925	30 (62.5%)	8 (16.7%)	0.652
IE	22 (44.0%)	45 (90%)	0.413	24 (48.9%)	47 (94.0%)	0.401	25 (50.0%)	4 (8.0%)	0.386
IR	7 (35.0%)	15 (75.0%)	0.247	7 (35%)	13 (65.0%)	0.406	8 (40.0%)	2 (1.0%)	0.491
MT	29 (63.0%)	34 (73.9%)	0.675	22 (47.8%)	37 (80.4%)	0.659	27 (58.7%)	10 (21.7%)	0.446
PP	12 (50.0%)	22 (91.7%)	0.523	11 (45.8%)	23 (95.8%)	0.516	10 (41.7%)	10 (41.7%)	0.335
QA	16 (40.0%)	32 (80.0%)	0.288	18 (45.0%)	38 (95.0%)	0.296	19 (47.5%)	3 (7.5%)	0.513
RC	24 (46.2%)	42 (80.8%)	0.377	23 (44.2%)	41 (78.8%)	0.440	27 (51.9%)	11 (21.2%)	0.469
all	148 (52.9%)	210 (75.0%)	0.522	146 (52.1%)	216 (77.1%)	0.561	146 (52.1%)	48 (17.1%)	0.425

Table 6.5: Accuracy and CWS for all three algorithms when testing on development set for threshold 0.5.

of a certain type. We choose, however, not to do so and to evaluate the model for each of the three algorithms separately.

6.3.2 Training Versus Given Threshold

To answer the question whether the dataset at hand is suitable for training, we repeat the experiments above, but now with a given threshold of 0.5 instead of an optimal one from training.

If we compare the Tables above with the results for threshold 0.5, (see Table 6.5), we see that accuracy and CWS are sometimes higher, sometimes lower for threshold 0.5. It seems that every subset has its own optimal threshold, which can be higher or lower than 0.5. Performance on the total test set is around performance for the optimal threshold from training.

However, these results do not give a good answer to the question whether training is useful. It could have been that not 0.5 but 0.45 gives a better result than training. Training would not be a good idea if training- and test set were very different and had other optimal

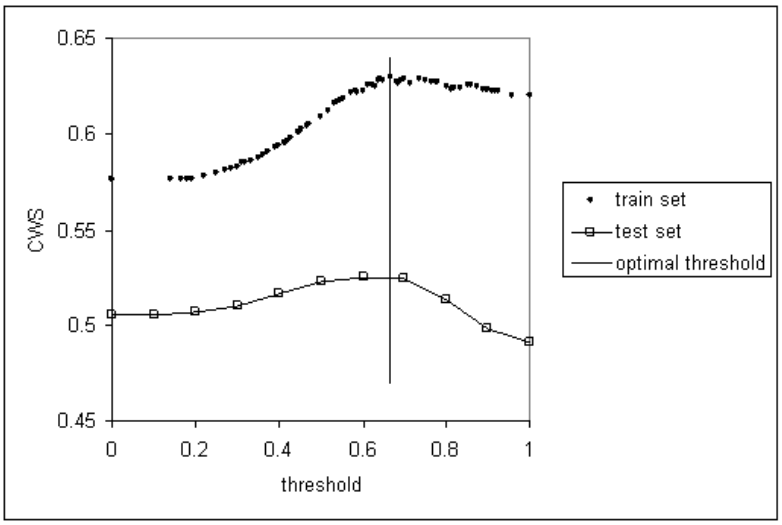


Figure 6.4: CWS for the development training- and test set and optimal threshold for the training set for the plain word overlap algorithm.

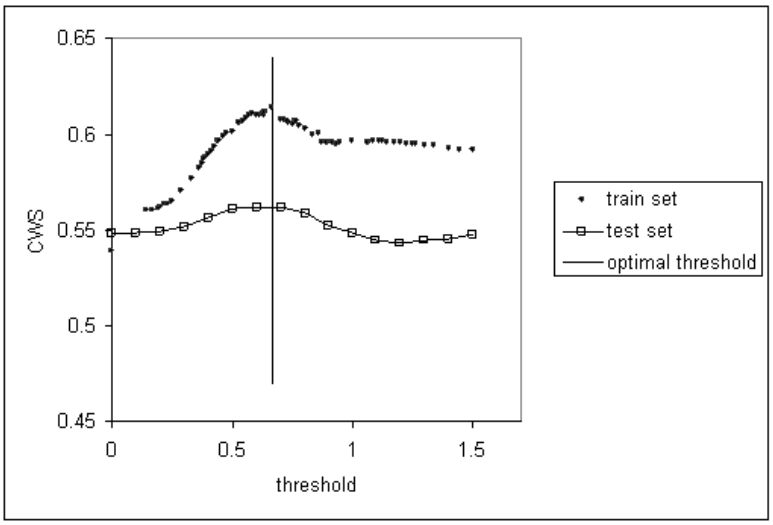


Figure 6.5: CWS for the development training- and test set and optimal threshold for the training set for the lemmatised word overlap algorithm.

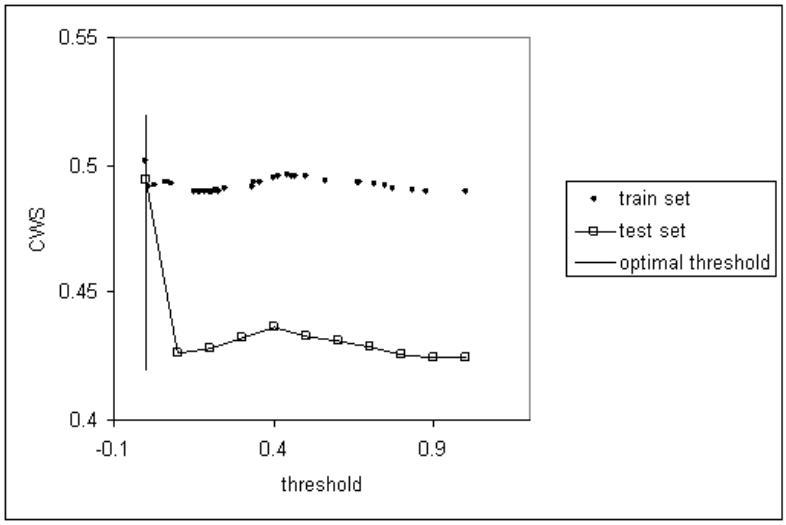


Figure 6.6: CWS for the development training- and test set and optimal threshold for the training set for the conceptual distance algorithm.

thresholds. We perform an experiment to find out how similar these development sets perform. We take a series of threshold values, 0.1 apart, for which we calculate the CWS on the evaluation test set. In Table 6.4, 6.5 and 6.6 the results of this experiment are displayed against the CWS of the development training set, for each of the three algorithms respectively. We see that the optimum is approximately the same for both training- and test set, indicating that the sets are not very different in nature and usable for training. That is, for the first two algorithms. The third one has an optimum at around 0, indicating that the model is poor. The fact that optimising the threshold does not seem to have much effect we think is due to the poor model, causing only a small range of CWS scores, more than to the lack of effect of the optimising.

6.4 Evaluation of the Model

We evaluate the model by training it, for each of the algorithms separately, on the development training set and testing it on the final 800-pair large evaluation set that was provided with the challenge. We also train and evaluate for each of the subsets separately. In 6.6, we see that the seven subsets are not evenly distributed. Because they all contain 50% true pairs, they all have a baseline of 50%.

Table 6.7, 6.8 and 6.9 show the results on the evaluation set using the thresholds we optimised on the training set *dev*. We see that the two word overlap algorithms perform better than the conceptual distance measure for the total data set, where the different subsets show a different picture; they all have a different algorithm that performs best on

	Evaluation set	
	Occurrences	Baseline
CD	150	50.0%
IE	120	50.0%
IR	90	50.0%
MT	120	50.0%
PP	50	50.0%
QA	130	50.0%
RC	140	50.0%
all	800	50.0%

Table 6.6: Amount of T-H pairs in and baseline for final evaluation set.

	threshold	accuracy	judged TRUE	CWS
CD	0.611011	122 (81.3%)	63 (42.0%)	0.924
IE	0.9999	52 (43.3%)	12 (1.0%)	0.412
IR	0.9999	41 (45.6%)	6 (6.7%)	0.371
MT	0.681718	55 (45.8%)	47 (39.2%)	0.483
PP	0.714186	25 (50.0%)	36 (72.0%)	0.458
QA	0.636264	62 (47.7%)	69 (53.1%)	0.370
RC	0.9999	63 (45.0%)	21 (15.0%)	0.405
all	0.666567	436 (54.5%)	436 (54.5%)	0.531

Table 6.7: Accuracy and CWS for plain word overlap algorithm when training on the development training set and testing on the evaluation set. Accuracy is expressed as number of correct judgements and percentage of correct judgements in brackets.

them. Figure 6.7 and 6.8 show that these preferences are dependent on whether we define performance to be accuracy or to be CWS.

Performance on the total evaluation set is 54.5%/0.531 for plain word overlap, 53.8%/0.570 for lemmatised word overlap and 50.0%/0.526 for conceptual distance.

We also perform evaluation with a fixed threshold of 0.5, and see that overall performance is around performance on a trained threshold for the two overlap measures, on conceptual distance it performs slightly better.

We expect performance for the total evaluation set to increase if we use an optimal threshold for every subset instead of an optimal threshold for all pairs together, but, against our expectation, we get CWS scores of 0.525, 0.529 and 0.485 respectively, which are lower.

We suspect that the poor performance of the conceptual distance algorithm is caused mainly by the one-to-one alignment of the assertions as was presented in Section 5.4.1.

	threshold	accuracy	judged TRUE	CWS
CD	0.4999	111 (74.0%)	90 (60.0%)	0.872
IE	0.2499	60 (50.0%)	120 (100%)	0.482
IR	0.9999	44 (48.9%)	13 (14.4%)	0.419
MT	0.7499	59 (49.2%)	53 (44.2%)	0.498
PP	0.9999	25 (50.0%)	14 (28.0%)	0.541
QA	0.666567	55 (42.3%)	76 (58.5%)	0.432
RC	0.3749	69 (49.3%)	131 (93.6%)	0.486
all	0.666567	430 (53.8%)	486 (60.8%)	0.570

Table 6.8: Accuracy and CWS for lemmatised word overlap algorithm when training on the development training set and testing on the evaluation set. Accuracy is expressed as number of correct judgements and percentage of correct judgements in brackets.

	threshold	accuracy	judged TRUE	CWS
CD	0.083233	96 (64.0%)	93 (62.0%)	0.682
IE	0.166567	66 (55.0%)	50 (41.7%)	0.525
IR	-0.0001	37 (41.1%)	58 (64.4%)	0.359
MT	-0.0001	54 (45.0%)	84 (70.9%)	0.422
PP	0.0624	26 (52.0%)	41 (82.0%)	0.503
QA	-0.0001	50 (38.5%)	69 (53.1%)	0.342
RC	-0.0001	59 (42.1%)	85 (60.7%)	0.407
all	-0.0001	400 (50.0%)	800(100%)	0.526

Table 6.9: Accuracy and CWS for conceptual distance algorithm when training on the development training set and testing on the evaluation set. Accuracy is expressed as number of correct judgements and percentage of correct judgements in brackets.

	plain word overlap			lemmatised word overlap			conceptual distance		
	accuracy	judged TRUE	CWS	accuracy	judged TRUE	CWS	accuracy	judged TRUE	CWS
CD	118 (78.7%)	77 (51.3%)	0.924	113 (75.3%)	84 (56.0%)	0.870	94 (62.7%)	27 (18.0%)	0.678
IE	64 (53.3%)	100 (83.3%)	0.474	59 (49.2%)	103 (85.8%)	0.460	64 (53.3%)	14 (11.7%)	0.515
IR	46 (51.1%)	59 (65.6%)	0.377	45 (50.0%)	54 (60.0%)	0.454	48 (53.3%)	9 (1.0%)	0.342
MT	55 (45.8%)	89 (74.2%)	0.509	58 (48.3%)	90 (75.0%)	0.512	60 (50.0%)	20 (16.7%)	0.444
PP	25 (50.0%)	48 (96.0%)	0.435	24 (48.0%)	49 (98.0%)	0.478	24 (48.0%)	25 (50.0%)	0.485
QA	61 (46.9%)	88 (67.7%)	0.388	58 (44.6%)	87 (66.9%)	0.443	61 (46.9%)	22 (16.9%)	0.351
RC	65 (46.4%)	119 (85.0%)	0.441	63 (45.0%)	119 (85.0%)	0.486	66 (47.1%)	25 (17.9%)	0.409
all	434 (54.3%)	580 (72.5%)	0.525	419 (52.4%)	587 (73.4%)	0.571	417 (52.1%)	143 (17.9%)	0.478

Table 6.10: Accuracy and CWS for all three algorithms when testing on evaluation set for threshold 0.5.

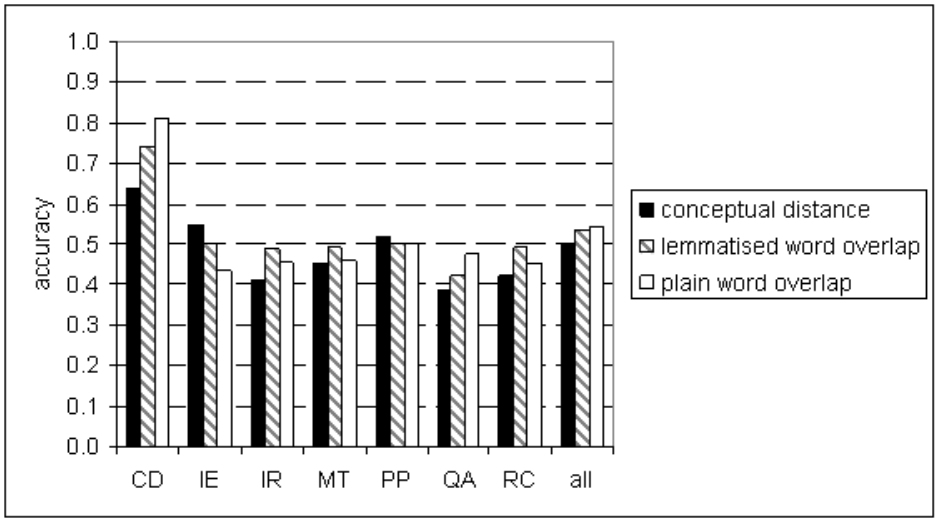


Figure 6.7: Accuracy (on a scale from 0 to 1) for each of the subsets trained separately and for the total evaluation set for al three algorithms.

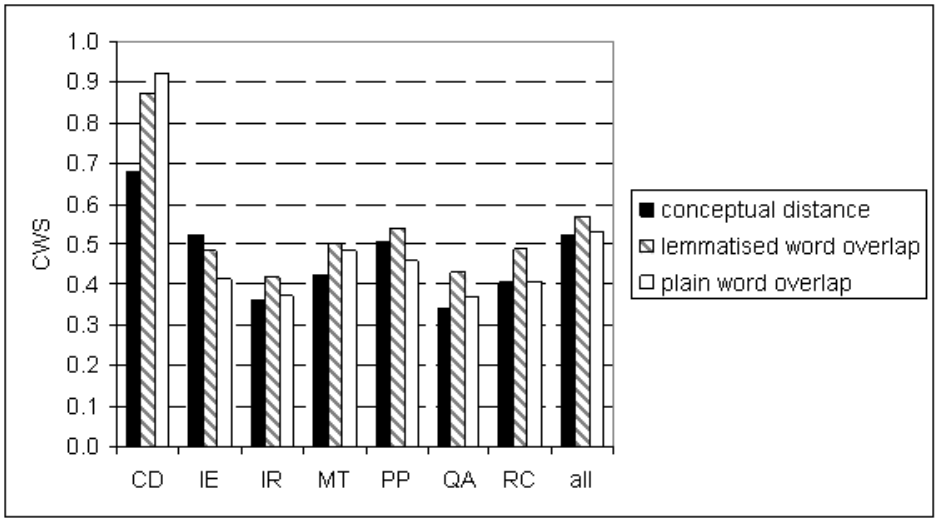


Figure 6.8: CWS for each of the subsets trained separately and for the total evaluation set for all three algorithms.

6.5 Other Approaches

16 Groups participated to the RTE1 challenge, submitting the results of a one run for every of a total of 28 different systems. Table 6.11 (taken from [DGM05]), gives its results and an indication of what components the different systems consist of. Popular components are different kinds of word overlap measures, the calculation of word-word relations based on statistical methods or WordNet, and matching of syntactic structures. Finally, a few systems used world knowledge and some used logical inference over semantically interpreted representations. Results for all of the runs are not very encouraging; accuracy is rarely above 0.6 and sometimes even under the baseline of 0.5, cws between 0.5 and 0.7. [DGM05] points at the fact that runs with cws>0.540 or accuracy>0.535 are better than chance at the 0.05 level and runs with cws>0.558 or accuracy>0.546 are better than chance at the 0.01 level. Some of the participants (for example [BM05] point out that the performance should, next to cws and accuracy, be measured in terms of precision, recall and f, in order to distinguish between runs with same accuracy or recall but different true/false positive/negative results. "All systems did not perform significantly better than the f=0.67 baseline of a system which uniformly predicts *true*"[DGM05]. The results for the CD subtask are generally significantly higher than performance on the other tasks (accuracy up to 87% and cws up to 0.95). For some systems, the success on the CD tasks pulls up the figures from the insignificance baselines [DGM05].

The approach of Jijkoun & De Rijke [JdR05], [JdR06] is comparable to our two word overlap measures. They calculate the directed sentence similarity by calculating "semantic" word overlap between text and hypothesis. Entailment is recognized by comparing the directed sentence similarity against a threshold, which is set by optimising accuracy on the total development set. As a measure for lexical similarity, they use Lin's dependency-based word similarity and a measure which is based on lexical chains in WordNet (see [BH01]). The similarity value for two words is weighted using normalised inverse collection frequency, calculated on a big collection of newspapers. The evaluation results for this system are comparable to ours. Performance on the evaluation set is worse than on the development corpus, optimal threshold and performance differ substantially for each of the subsets, very well on CD but poor on the rest. Optimising the threshold can be used to fine-tune the precision/recall balance, but only very limited due to overfitting.

Bos & Markert [BM05] have a different approach, combining a shallow word overlap method with a deep semantic analysis using theorem proving techniques. They train a decision tree on the results for the development set. Their word overlap measure differs from ours as it uses inverse document frequency over the Internet as a weight which is added if a word in the Hypothesis is present in the Text or subtracted otherwise. Word are lemmatise first, like we do. Another difference is the use of 10-fold cross validation on the total development set, where we use the leave-one-out method. The deep seman-

tic analysis is done using Hockenmaier's StatCCG to generate semantic representations, as is described in [BCS+04]. The semantic representation language is a first-order fragment of Discourse Representation Theory. Bos & Markert use two different first-order logic theorem provers for finding proofs. To support the proofs, they constructed a background knowledge database, containing axioms for the semantics of different semantic constructions, WordNet hypernym information for the text and hypothesis, and geographical information.

Notwithstanding the complexity of the system, performance on the evaluation set was only slightly better than baseline on the 1% level (accuracy: 0.562/CWS: 0.591 and only slightly better than the shallow word overlap measure alone. This is due to the fact that the model is reasonably accurate on only a very small part of the data; coverage is small.

First author (Group)	accuracy	cws	partial coverage	System description					
				Word overlap	Statistical lexical relations	WordNet	Syntactic matching	World knowledge	Logical inference
Akhmatova (Macquarie)	0.519	0.507		X					X
Andreevskaia (Concordia)	0.519	0.515				X	X		
	0.516	0.52							
Bayer (MITRE)	0.586	0.617			X				
	0.516	0.503	73%					X	X
Bos (Edinburgh & Leeds)	0.563	0.593		X		X		X	X
	0.555	0.586		X					
Delmonte (Venice & irst)	0.606	0.664	62%			X	X		X
Fowler (LCC)	0.551	0.56				X		X	X
Glickman (Bar Ilan)	0.586	0.572			X				
	0.53	0.535							
Herrera (UNED)	0.566	0.575		X	X		X		
	0.558	0.571		X					
Jijkoun (Amsterdam)	0.552	0.559		X	X				
	0.536	0.553		X		X			
Kouylekov (irst)	0.559	0.607		X	X		X		
	0.559	0.585							
Newman (Dublin)	0.563	0.592		X	X				
	0.565	0.6							
Perez (Madrid)	0.495	0.517		X					
	0.7	0.582	19%						
Punyakanok (UIUC)	0.561	0.569					X		
Raina (Stanford)	0.563	0.621			X	X	X		X
	0.552	0.686							
Wu (HKUST)	0.512	0.55			X		X		
	0.505	0.536							
Zanzotto (Rome-Milan)	0.524	0.557				X	X		
	0.518	0.559							

Table 6.11: Accuracy and cws results for the system submissions, ordered by first author. Partial cover- age refers to the percentage of examples classified by the system out of the 800 test examples. Table taken from [DGM05].

Chapter 7

Conclusions

In this thesis, we described a method for recognising textual entailment by calculating the distance between two extracted semantic representations containing head verbs, their arguments and semantic roles, negations and sentence modifiers. For the semantic representations that were structurally near, we calculated conceptual distance between the terms with the same semantic role, using conceptual context as a measure for conceptual distance.

Accuracy for our model was just above baseline for a threshold of 0.5, but CWS was low (0.478). For a threshold optimised on the development set, CWS was a bit higher (0.526), but accuracy was 50% and all pairs were judged TRUE. We can conclude from this that performance is poor on the development set, resulting in an optimal threshold of around 0 (random guessing), and that optimising the threshold does not work for a model with a performance as poor as this. Evaluation of the different subsets shows, however, good performance for some of the subsets and optimising the threshold seems beneficial for the performance for these subsets. Surprisingly, accuracy for the total evaluation set did not increase when optimising each subset individually. We did not test this for CWS, but this would be interesting to do.

We compared the performance of our model to the results of a basic word overlap model and a word overlap model enriched with PoS information and lemmatisation. They both outperformed the conceptual distance model, the enriched word overlap model having a CWS of 0.570. When looking at the different subsets, we see that each has a "favourite" algorithm. Combining the three measures may result in a higher performance.

The poor performance, we believe, is partly due to the alignment of the semantic representations before calculating conceptual distance, instead of calculating the conceptual distance of each pair of representations. We chose for this mechanism for run time reasons. We expect that improving the efficiency of the system would deal with this problem, and extensive conceptual distance calculation would then become possible and may increase

performance.

Evaluations of the other approaches to RTE showed similar results. As some solutions performed statistically better than random guessing, but not much, RTE is clearly a difficult task and still in its infancy. Some of the approaches, however, seem promising and the high interest from the NLP community in the task raises expectations.

Some of the participants to the challenge have had some doubts about the quality of the RTE data set. It also appeared to be somewhat obscure what definition of Textual Entailment the organisers of the challenge must have had. At least three groups have independently done manual annotations of portions of the data set and report annotator-agreement levels with the gold-standard data of 95% on the test set (see[DGM05]), which we think is not low at all.

Zaenen, Karttunen and Crouch [ZKC05] started an interesting discussion about the definition of the task of Textual Entailment, or Local Textual Inference as they like to call it to avoid confusion with logical entailment. Note that in Textual Entailment, it is possible to go beyond the set of actual premises and conclude things that are very reasonable but not strictly logically entailed. Zaenen, Karttunen and Crouch distinguish three types of inference: (logical) entailment, conventional implicatures (including presuppositions) and conversational implicatures. They accept the first two types as local textual inference, but have difficulties with the third as it often expresses speaker meaning instead of reliable facts. They also mention examples that, in their view, should not belong to the task or are erroneous and argue that it would be better to distinguish between inferences that are based only on linguistic knowledge and those also based on world knowledge, and between strictly valid sentences and merely plausible inferences. Manning [Man06], however, sees this as a attempt to narrow the definition of RTE and to exclude many inferences that a common man would make when reading a sentence and that would be needed for robust interpretation. Zaenen et al. [KCZ06] defend their position arguing that they only like to circumscribe the impact of world knowledge and the plausibility of the inference and not exclude it from the data set.

Many other interesting issues are being brought up in the discussion, e.g. whether annotator agreement would have been higher if RTE had had a more formal definition, whether spelling variants and -mistakes should be part of the task or not, how we would define 'a common man', whether there is a clear boundary between linguistic, world- and common sense knowledge, and whether the task would have benefited from hand-crafted examples and whether these would be legitimate? These issues go beyond the scope of this thesis.

Appendix A

Penn Treebank Tag Set

Source: M.P Marcus et al. [MSM94].

CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential ‘there’
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun
PRP$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	‘to’

UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun
WP$	Possessive wh-pronoun
WRB	Wh-adverb
#	Pound sign
$	Dollar sign
.	Sentence-final punctuation
,	Comma
:	Colon, semi-colon
(	Left bracket character
)	Right bracket character
"	Straight double quote
`	Left open single quote
"	Left open double quote
’	Right close single quote
"	Right close double quote

Bibliography

[BB05] P. Blackburn and J. Bos. *Representation and Inference for Natural Language. A First Course in Computational Semantics*. CSLI, 2005.

[BCS+04] J. Bos, S. Clark, M. Steedman, J.R. Curran, and J. Hockenmaier. Wide-coverage semantic representations for a ccg parser. In *Proceedings of the 20th International Conference on Computational Linguistics*, volume II. Coling, 2004. http://web.comlab.ox.ac.uk/oucl/work/stephen.clark/papers/coling_bos04.pdf.

[BH53] Y. Bar-Hillel. A quasi-arithmetical notation for syntactic description. *Language*, 29:47–58, 1953.

[BH01] A. Budanitsky and G. Hirst. Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures. In *Proceedings of the NAACL '01 Workshop on WordNet and Other Lexical Resources, in the North American Chapter of the Association for Computational Linguistics (NAACL-2001), Pittsburgh, USA*, 2001. http://ftp.cs.toronto.edu/pub/gh/Budanitsky+Hirst-2001.pdf.

[BM05] J. Bos and K. Markert. Combining shallow and deep methods for recognizing textual entailment. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, 2005. http://www.cs.biu.ac.il/~glikmao/rte05/rte05_proceedings.pdf.

[Bos05] J. Bos. Towards wide-coverage semantic interpretation. In *Proceedings of IWCS-6*, 2005. http://www.iccs.informatics.ed.ac.uk/~jbos/pubs/bos-iwcs-6.pdf.

[Bra00] T. Brants. Tnt – a statistical part-of-speech tagger. In *Proceedings of the 6th Applied NLP Conference, ANLP-2000, April 29 – May 3, 2000, Seattle, WA.*, 2000. http://www.coli.uni-saarland.de/~thorsten/publications/Brants-TR-TnT.pdf.

[CMU+97] R. Cole, J. Mariani, H. Uszkoreit, A. Zaenen, and V. Zue. Survey of the state of the art in human language technology, 1997. http://www.lt-world.org/HLT_Survey/master.pdf.

[Col99] M. Collins. *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania, 1999. http://acl.ldc.upenn.edu/J/J03/J03-4003.pdf.

[Con96] FRACAS Consortium. Using the framework. In *Fracas project LRE 62-051. Deliverable D16*, 1996. ftp://ftp.cogsci.ed.ac.uk/pub/FRACAS/del16.ps.gz.

[DGM05] I. Dagan, O. Glickman, and B. Magnini. The pascal recognising textual entailment challenge. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, 2005. http://www.cs.biu.ac.il/~glikmao/rte05/rte05_proceedings.pdf.

[GH03] D. Gildea and J. Hockenmaier. Identifying semantic roles using Combinatory Categorial Grammar. In *2003 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 57–64, Sapporo, Japan, 2003. http://www.cs.rochester.edu/~gildea/gildea-emnlp03.pdf.

[GP01] D. Gildea and M. Palmer. The necessity of parsing for predicate argument recognition. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, Annual Meeting of the ACL, Philiadelphia, 2001. http://www.egr.msu.edu/~jchai/QAPapers/gildea-acl02.pdf.

[Hoc03] J. Hockenmaier. *Data and Models for Statistical Parsing with Combinatory Categorial Grammar*. PhD thesis, School of Informatics, University of Edinburgh, 2003. http://www.ircs.upenn.edu/~juliahr/Dissertation/thesis.pdf.

[JdR05] V. Jijkoun and M. de Rijke. Recognizing textual entailment using lexical similarity. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, 2005. http://www.cs.biu.ac.il/~glikmao/rte05/rte05_proceedings.pdf.

[JdR06] V. Jijkoun and M. de Rijke. Recognizing textual entailment: Is lexical similarity enough? In I. Dagan, F. Dalche, J. Quinonero Candela, and B. Magnini, editors, *Evaluating Predictive Uncertainty, Textual Entailment and Object Recognition Systems*, volume 3944 of *LNAI*, pages 449–460. Springer, April 2006. http://www.science.uva.nl/~mdr/Publications/Files/pascal-2005-rte-proceedings.pdf.

[KCZ06] L. Karttunen, R. Crouch, and A. Zaenen. Circumscribing is not excluding: A reply to manning. http://www2.parc.com/istl/members/karttune/publications/reply-to-manning.pdf, 2006.

[KP02] P. Kingsbury and M. Palmer. From treebank to propbank. In *Proceedings of the LREC, Las Palmas, Canary Islands, Spain*, 2002.

[LLSB04] H. Lieberman, H. Liu, P. Singh, and B. Barry. Beating common sense into interactive applications. *Artificial Intelligence Magazine*, 25(4):63–76, 2004. http://web.media.mit.edu/~lieber/Lieberary/Common-Sense/Beating-Common-Sense/Beating-Common-Sense.pdf.

[LS04a] H. Liu and P. Singh. Commonsense reasoning in and over natural language. In *Proceedings of the 8th International Conference on Knowledge-Based Intelligent Information & Engineering Systems (KES'2004). Wellington, New Zealand. September 22-24.*, Lecture Notes in Intelligence, page 9. Springer, 2004. http://web.media.mit.edu/~push/CommonsenseInOverNL.pdf.

[LS04b] H. Liu and P. Singh. Conceptnet: A practical commonsense reasoning toolkit. *BT Technology Journal*, 22(4), 2004. http://web.media.mit.edu/~hugo/publications/papers/BTTJ-ConceptNet.pdf.

[Man06] C.D. Manning. Local textual inference: it's hard to circumscribe, but you know it when you see it - and nlp needs it. http://nlp.stanford.edu/~manning/papers/LocalTextualInference.ps, February 2006.

[MSM94] M.P. Marcus, B. Santorini, and M.A. Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330, 1994. http://acl.ldc.upenn.edu/J/J93/J93-2004.pdf.

[Ste00] M. Steedman. *The Syntactic Process*. The MIT Press, Cambridge, Massachusetts, 2000.

[ZKC05] A. Zaenen, L. Karttunen, and R. Crouch. Local textual inference: can it be defined or circumscribed? In *Proceedings of the ACL 2005 Workshop on Emprirical Modelling of Semantic Equivalence and Entailment*, 2005. http://www2.parc.com/istl/members/zaenen/publications/acl2005workshop.pdf.