Recognising Textual Entailment with Robust Logical Inference

Johan Bos[1] and Katja Markert[2]

[1] School of Informatics, University of Edinburgh, UK
jbos@inf.ed.ac.uk
[2] School of Computing, University of Leeds, UK
markert@comp.leeds.ac.uk

Abstract. We use logical inference techniques for recognising textual entailment, with theorem proving operating on deep semantic interpretations as the backbone of our system. However, the performance of theorem proving on its own turns out to be highly dependent on a wide range of background knowledge, which is not necessarily included in publically available knowledge sources. Therefore, we achieve robustness via two extensions. Firstly, we incorporate model building, a technique borrowed from automated reasoning, and show that it is a useful robust method to approximate entailment. Secondly, we use machine learning to combine these deep semantic analysis techniques with simple shallow word overlap. The resulting hybrid model achieves high accuracy on the RTE testset, given the state of the art. Our results also show that the various techniques that we employ perform very differently on some of the subsets of the RTE corpus and as a result, it is useful to use the nature of the dataset as a feature.

1 Introduction

Recognising textual entailment (RTE) is the task to find out whether some text T entails a hypothesis text H. This task has recently been the focus of the *RTE challenge*, an evaluation exercise organised by the PASCAL network in 2004/05 [DGM05]. Two examples from the dataset issued as part of this challenge are shown below.[1] In Example 1550 H follows from T (hence it is marked TRUE) whereas this is not the case in Example 731 (which is therefore annotated as FALSE).

Example: 1550 (TRUE)

T: In 1998, the General Assembly of the Nippon Sei Ko Kai (Anglican Church in Japan) voted to accept female priests.

H: The Anglican church in Japan approved the ordination of women.

[1] All examples in this article are from the corpus released as part of the RTE challenge, keeping the original example identifiers. This corpus is available via `http://www.pascal-network.org/Challenges/RTE/`.

Example: 731 (FALSE)

T: The city Tenochtitlan grew rapidly and was the center of the Aztec's great empire.

H: Tenochtitlan quickly spread over the island, marshes, and swamps.

The recognition of textual entailment needs access to a wide range of syntactic and lexical knowledge and is without doubt one of the ultimate challenges for any natural language processing (NLP) system: if it is able to recognise entailment with reasonable accuracy, it is clearly an indication that it has some thorough understanding of how language works. Moreover, recognising entailment bears similarities to Alan Turing's famous test to assess whether machines can think, as access to different sources of knowledge and the ability to draw inferences seem to be among the primary ingredients for an intelligent system. In addition, many NLP tasks have strong links to entailment: in summarisation, a summary should be entailed by the text; in machine translation, the source and target text should mutually entail each other; in question answering, a valid answer entails the question; and in information extraction, the extracted information should also be entailed by the text.

In this article we discuss two methods for recognising textual entailment. Firstly, we present a shallow method that relies mainly on weighted word overlap between text and hypothesis. Secondly, we use a method based on deep semantic analysis, borrowing techniques from the field of automated deduction, namely theorem proving and model building, to perform logical inference. As theory for natural language semantics we use Discourse Representation Theory, DRT [KR93]. Semantic representations are built in a compositional way for the text and hypothesis with the help of Combinatory Categorial Grammar, CCG [Ste01], using a wide-coverage statistical parser [CC04].

To increase transferrability to other datasets and tasks, both the shallow and deep method are domain-independent, and neither has been tailored to any particular test suite. In this article we test their accuracy and robustness on the RTE datasets as one of the few currently available datasets for textual inference. We also combine the two methods in a hybrid approach using an off-the-shelf machine learning tool.

In particular, we are interested in the following questions:

– Can either of the methods presented improve significantly over the baseline and what are the performance differences between them?
– How far does deep semantic analysis suffer from a lack of lexical and world knowledge and how can we perform robust logical inference in the face of potentially large knowledge gaps?
– Does the hybrid system using both shallow and deep semantic analysis improve over the individual use of these methods?
– How does the design of the test suite affect performance? Are there subsets of the test suite that are more suited to any particular textual entailment recognition method?

This article is organised as follows. First, we will describe the shallow semantic approach (Section 2). This is followed by a detailed description of the deep semantic approach in Section 3, where we will also explain how to integrate theorem proving and model building, and how this combination achieves a level of robustness in using logical inference. Both Sections 2 and 3 list the features derived from these methods that are then used in several machine learning experiments. Section 4 describes these experiments, including a hybrid deep/shallow system for entailment recognition. In Section 5 we perform an error analysis, and Section 6 discusses the RTE challenge in general. Finally, related work is presented in Section 7.

2 Shallow Semantic Analysis

We use several shallow surface features to model the text, hypothesis and their relation to each other. A basic shallow feature we believe might play a role in textual entailment is word overlap.

2.1 Word Overlap Measures

For many textual entailment example pairs there is a dependency between surface string similarity of text and hypothesis and the existence of entailment. In particular, the inclusion of all or almost all words of the hypothesis in the text makes entailment likely. A case in point is Example 125 below, where all words in the hypothesis are contained in the text and entailment holds.

Example: 125 (TRUE)

T: On November 25, 2001, a baby Asian elephant was born at the National Zoo in Washington, DC.
H: baby elephant born

In contrast, the occurrence of words in the hypothesis that are unrelated to any word in the text makes entailment unlikely. This is the case in Example 731 in the Introduction where the words *swamps*, *islands* and *marshes* in the hypothesis introduce new information that is not in the text.

Our general model for measuring word overlap is a simple bag-of-words model, i.e. word order and syntactic structure are ignored. The exact procedure is as follows:

1. Both text and hypothesis are tokenised and lemmatised.
2. The overlap measure between text and hypothesis is initialised as zero.
3. Should a lemma in the hypothesis be *related to* a lemma in the text, its *weight* is added to the overlap measure, otherwise it is ignored.
4. In the end the overlap is normalised by dividing it by the sum of the weights of all lemmas in the hypothesis. This ensures that the overlap is always a real number between 0 and 1 and also ensures independence of the length of the hypothesis.

Variations of this simple measure depend on the definition of word relatedness as well as on which formula for weighting lemmas is used. Thus, it is possible to define relatedness simply as equality of lemmas. This can handle examples like 125 above well, where all lemmas in the hypothesis are exactly matched in the text. However, it falls short of recognising, for example, simple synonym variation in word choice as the replacement of *found* by *discovered* in Example 1987 below.

Example: 1987 (TRUE)

T: The girl was found in Drummondville earlier this month.

H: The girl was discovered in Drummondville.

Therefore the word overlap we use takes into account equality, synonymy and morphological derivations. WordNet [Fel98] is used as the knowledge source for synonymy and derivations. We therefore say that a lemma l_1 in the hypothesis is said to be *related* to a lemma l_2 in the text iff l_1 and l_2 are equal, belong to the same WordNet synset (e.g., "murder" and "slay"), are related via WordNet derivations (e.g. "murder" and "murderer") or are related via a combination of synonymy and derivations (e.g. "murder" via "murderer" to "liquidator"). No word sense disambiguation is performed and *all* synsets for a particular lemma are considered.

Regarding weight assignment, each lemma in the hypothesis is assigned its inverse document frequency, accessing the Web as corpus via the GoogleAPI, as its weight. We prefer this procedure over equal weighting of all lemmas as it allows us to assign more importance to less frequent words.

2.2 Shallow Semantic Features

Apart from the overlap measure `wnoverlap` we take into account length (as measured by number of lemmas) of text and hypothesis, because in most true entailments the hypothesis is shorter than the text as it contains less information. This is covered by three numerical features measuring the length of the text, of the hypothesis and the relative length of hypothesis with regard to the text.

3 Deep Semantic Analysis

3.1 Overview

In a nutshell, the deep semantic analysis aims to produce fine-grained semantic representations for both the text and hypothesis of an entailment pair. It then uses techniques from automated deduction, in particular first-order theorem proving and finite model building, to predict whether the text entails the hypothesis or not.

To achieve this, both the text and hypothesis are tokenised using NLProcesser[2] and fed into a robust wide-coverage parser, which uses a statistical

[2] A product from Infogistics, see `http://www.infogistics.com/textanalysis.html`

model based on CCG [CC04]. The parser produces CCG-derivations. Consider for instance the derivations for Example 78 below:

```
Clinton      'S        new  book        is               not          big  seller        here           .
------- ------------- ----- ----- -------------- ------------------- ----- ------ ------------------- -----
   N    (NP[nb]/N)\NP  N/N    N   (S[dcl]\NP)/NP (S[X]\NP)\(S[X]\NP)  N/N    N    (S[X]\NP)\(S[X]\NP)   .
-------               ----------> -------------------------------<Bx ---------->
  NP                       N                (S[dcl]\NP)/NP                N
 -------------------<                                                ----------- 
       NP[nb]/N                                                          NP
      ---------------------------> -------------------------------------------->
                NP[nb]                             S[dcl]\NP
                                                  ---------------------------------------------------<
                                                                       S[dcl]\NP
      ----------------------------------------------------------------------------<
                                         S[dcl]
                                        ------------------------------------------------------------------>P
                                                                      S[dcl]
```

```
Clinton      's        book         is         a      big  seller   .
------- ------------- ----- -------------- -------- ----- ------ -----
   N    (NP[nb]/N)\NP   N   (S[dcl]\NP)/NP NP[nb]/N  N/N    N      .
-------                                             ---------->
  NP                                                      N
 -------------------<                      ---------------->
       NP[nb]/N                                 NP[nb]
      ---------------------> ---------------------------->
             NP[nb]                 S[dcl]\NP
            ------------------------------------<
                           S[dcl]
                          -------------------------------------------->P
                                             S[dcl]
```

The CCG-derivations produced by the parser are the basis to construct discourse representation structures (DRSs, the semantic representations from DRT). This is done in a compositional way using the lambda-calculus as a "glue" to connect CCG with DRT [BCS$^+$04, Bos05]. The semantic representations are then translated into first-order logic expressions. This allows us to perform inferences using a general-purpose theorem prover and model builder.

3.2 Semantic Interpretation

The semantic representation language is a first-order fragment of the DRS-language used in Discourse Representation Theory [KR93], conveying argument structure with a neo-Davidsonian analysis and including the recursive DRS structure to cover negation, disjunction, and implication. A basic DRS is an ordered pair of a set of discourse referents and a set of conditions imposed on these discourse referents. For convenience, we use the boxed notation to visualise DRSs, with the discourse referents at the top and the conditions at the lower part of the box. Consider for example the following entailment pair and the DRSs constructed for it:

Example: 78 (FALSE)

T: Clinton's new book is not big seller here.

H: Clinton's book is a big seller.

$$d(\mathrm{T}): \begin{array}{|c|}\hline \mathrm{x1\ x2\ x3} \\ \hline \mathrm{book(x1)} \\ \mathrm{book(x2)} \\ \neg \begin{array}{|c|}\hline \ \\ \hline \mathrm{x1{=}x2} \\ \hline \end{array} \\ \mathrm{clinton(x3)} \\ \mathrm{of(x1,x3)} \\ \neg \begin{array}{|c|}\hline \mathrm{e4\ x5} \\ \hline \mathrm{big(x5)} \\ \mathrm{seller(x5)} \\ \mathrm{be(e4)} \\ \mathrm{agent(e4,x1)} \\ \mathrm{patient(e4,x5)} \\ \mathrm{here(e4)} \\ \hline \end{array} \\ \hline \end{array} \qquad d(\mathrm{H}): \begin{array}{|c|}\hline \mathrm{x1\ x2\ e3\ x4} \\ \hline \mathrm{book(x1)} \\ \mathrm{clinton(x2)} \\ \mathrm{of(x1,x2)} \\ \mathrm{big(x4)} \\ \mathrm{seller(x4)} \\ \mathrm{be(e3)} \\ \mathrm{agent(e3,x1)} \\ \mathrm{patient(e3,x4)} \\ \hline \end{array}$$

Proper names and definite descriptions are treated as anaphoric, and bound to previously introduced discourse referents if possible, otherwise accommodated [VdS92, Bos03]. Some lexical items are specified as presupposition triggers. An example is the adjective *new* which has a presuppositional reading, as shown by the existence of two different "book" entities in d(T). Scope is fully specified.

The DRS language, with its quantifier free and rather flat representations, is useful for resolving ambiguities, such as pronoun resolution and presupposition projection. However, no efficient reasoning engines that work directly on DRSs exist. Therefore we use a translation from DRSs to first-order formula syntax, which opens the doors for using first-order theorem proving technology [BBKdN01]. The literature on DRT offers several translation functions that map DRS onto first-order logic — here we apply the so-called standard translation from DRS to first-order logic (see e.g. [KR93, BBKdN01]). This function, f, is defined by the following clauses:

$$f(\begin{array}{|c|}\hline \mathrm{x}_1 \ldots \mathrm{x}_n \\ \hline \mathrm{C}_1 \\ \vdots \\ \mathrm{C}_n \\ \hline \end{array}) = \exists \mathrm{x}_1 \ldots \exists \mathrm{x}_n \ (f(\mathrm{C}_1) \wedge \ldots \wedge f(\mathrm{C}_n))$$

$$f(\begin{array}{|c|}\hline \mathrm{x}_1 \ldots \mathrm{x}_n \\ \hline \mathrm{C}_1 \\ \vdots \\ \mathrm{C}_n \\ \hline \end{array} \Rightarrow \mathrm{B}) = \forall \mathrm{x}_1 \ldots \forall \mathrm{x}_n \ ((f(\mathrm{C}_1) \wedge \ldots \wedge f(\mathrm{C}_n)) \rightarrow f(\mathrm{B}))$$

$f(\mathrm{B}_1 \vee \mathrm{B}_2) = f(\mathrm{B}_1) \vee f(\mathrm{B}_2)$
$f(\neg\ \mathrm{B}) = \neg\ f(\mathrm{B})$
$f(\mathrm{x}{=}\mathrm{y}) = \mathrm{x}{=}\mathrm{y}$

$f(\mathrm{P}(\mathrm{x})) = \mathrm{P}(\mathrm{x})$
$f(\mathrm{R}(\mathrm{x,y})) = \mathrm{R}(\mathrm{x,y})$

Here B, B_1 and B_2 are variables ranging over DRSs, C_i is a DRS-condition, P is a one-place predicate symbol and R a two-place predicate symbol. To illustrate the translation function, consider the result of translating d(T) and d(H) from Example 78 above:

$f(d(\mathrm{T}))$: ∃x1 ∃x2 ∃x3 (book(x1) ∧ book(x2) ∧ ¬ (x1=x2) ∧ clinton(x3) ∧ of(x1,x3) ∧ ¬ ∃e4 ∃ x5(big(x5) ∧ seller(x5) ∧ be(e4) ∧ agent(e4,x1) ∧ patient(e4,x5) ∧ here(e4)))

$f(d(\mathrm{H}))$: ∃x1 ∃x2 ∃e3 ∃x4 (book(x1) ∧ clinton(x2) ∧ of(x1,x2) ∧ big(x4) ∧ seller(x4) ∧ be(e3) ∧ agent(e3,x1) ∧ patient(e3,x4))

3.3 Theorem Proving

There are two kinds of first-order inference engines we use to perform reasoning: a theorem prover, and a model builder. First we will discuss the use of the theorem prover. We have integrated the prover Vampire [RV02] into our system, which is a general-purpose off-the-shelf theorem prover.

Given a textual entailment pair T/H, a theorem prover can be used to find an answer to conjecture (A):

$$f(d(\mathrm{T})) \rightarrow f(d(\mathrm{H})) \tag{A}$$

If the theorem prover manages to find a proof for this conjecture, then we predict that T entails H. (Note that we use the term “predict” here. There are cases in which the theorem prover finds a proof, although we are not actually dealing with a true entailment. This is due to inaccurate semantic analysis. See Section 5 for further discussion on this topic.)

In addition, we can also use a theorem prover to detect inconsistencies in a T/H pair by letting it handle the input (B):

$$\neg(f(d(\mathrm{T})) \wedge f(d(\mathrm{H}))) \tag{B}$$

If the theorem prover returns a proof for (B), we know that combining T and H yields an inconsistent state, thereby predicting that T does not entail H. An example is a pair T: *John is a doctor* and H: *John is not a doctor.* (Note that although it is the case that if T and H (combined) are inconsistent, then T does not entail H, the reverse does not hold. Therefore consistency checking is only a partial method to check for non-entailment.)

The RTE dataset contains only few inconsistent T/H pairs. Even although Example 78 might look like a case in point, it is not inconsistent. It would be if the T in the example were *Clinton's new book is not a big seller*. The addition of the adverb *here* makes T+H consistent.

Let's consider some examples that our system deals with successfully. Example 1005 is a case with apposition, and Example 898 one involving VP coordination.

Example: 1005 (TRUE)

T: Jessica Litman, a law professor at Michigan's Wayne State University, has specialized in copyright law and Internet law for more than 20 years.
H: Jessica Litman is a law professor.

Example: 898 (TRUE)

T: After the war the city was briefly occupied by the Allies and then was returned to the Dutch.
H: After the war, the city was returned to the Dutch.

Examples like these are rather trivial from the inference point of view, because they rely almost exclusively on correct syntactic analyses (here: apposition and coordination) and no additional knowledge is required to support the theorem prover. However, the majority of the entailment pairs require background knowledge to predict an entailment. In the next section we show what kind of background knowledge we use and how we integrate it.

3.4 Background Knowledge

To perform any interesting reasoning, the theorem prover needs background knowledge to support its proofs. For the RTE challenge we distinguished between three kinds of background knowledge: generic knowledge, lexical knowledge and geographical knowledge.

Knowledge is represented as axioms in first-order logic. Assume that BK is a conjunction of first-order axioms representing the relevant background knowledge for a T/H pair. The input to the theorem prover is then:

$$\text{BK} \wedge (f(d(\text{T})) \rightarrow f(d(\text{H}))) \qquad (\text{A}')$$

Generic Knowledge. Axioms for generic knowledge cover the semantics of possessives, active-passive alternation, and spatial knowledge. There are a dozen different axioms in the current system and these are the only manually generated axioms. Some examples include:

$\forall x \forall y \forall z(\text{in}(x,y) \wedge \text{in}(y,z) \rightarrow \text{in}(x,z))$
$\forall e \forall x \forall y(\text{event}(e) \wedge \text{agent}(e,x) \wedge \text{in}(e,y) \rightarrow \text{in}(x,y))$
$\forall e \forall x \forall y(\text{event}(e) \wedge \text{patient}(e,x) \wedge \text{in}(e,y) \rightarrow \text{in}(x,y))$
$\forall e \forall x \forall y(\text{event}(e) \wedge \text{theme}(e,x) \wedge \text{in}(e,y) \rightarrow \text{in}(x,y))$
$\forall x \forall y(\text{in}(x,y) \rightarrow \exists e(\text{locate}(e) \wedge \text{patient}(e,x) \wedge \text{in}(e,y)))$
$\forall x \forall y(\text{of}(x,y) \rightarrow \exists e(\text{have}(e) \wedge \text{agent}(e,y) \wedge \text{patient}(e,x)))$
$\forall e \forall x(\text{event}(e) \wedge \text{agent}(e,x) \rightarrow \text{by}(e,x))$

The last axiom in this list, for instance, helps Vampire to find a proof for Example 1977, which is a case of active-passive alternation.

<u>Example: 1977 (TRUE)</u>

T: His family has steadfastly denied the charges.

H: The charges were denied by his family.

$$d(\mathrm{T}): \begin{array}{|c|}\hline \text{e1 x2 x3 x4} \\ \hline \text{male(x4)} \\ \text{of(x3,x4)} \\ \text{family(x3)} \\ \text{charge(x2)} \\ \text{deny(e1)} \\ \text{agent(e1,x3)} \\ \text{patient(e1,x2)} \\ \text{steadfastly(e1)} \\ \hline \end{array} \qquad d(\mathrm{H}): \begin{array}{|c|}\hline \text{e1 x2 x3} \\ \hline \text{male(x3)} \\ \text{charge(x3)} \\ \text{deny(e1)} \\ \text{patient(e1,x3)} \\ \text{of(x2,x3)} \\ \text{family(x2)} \\ \text{by(e1,x2)} \\ \hline \end{array}$$

Lexical Knowledge. Lexical knowledge is created automatically from WordNet. A hyponymy relation between two synsets A and B is converted into ∀x(A(x)→B(x)). Two synset sisters A and B are translated into ∀x(A(x)→ ¬B(x)). Here the predicate symbols from the DRS are mapped to WordNet synsets using a variant of Lesk's WSD algorithm [MS99]. The aforementioned Example 78 would be supported by the following lexical axioms:

∀x(clinton(x)→person(x))
∀x(book(x)→artifact(x))
∀x(artifact(x)→ ¬person(x))

Consider for instance Example 1952 below. The axiom ∀x(soar(x)→rise(x)) suffices for finding a proof for this entailment pair:

<u>Example: 1952 (TRUE)</u>

T: Crude oil prices soared to record levels.

H: Crude oil prices rise.

$$d(\mathrm{T}): \begin{array}{|c|}\hline \text{e1 x2 x5 x6} \\ \hline \text{crude(x5)} \\ \text{oil(x6)} \\ \text{nn(x6,x5)} \\ \text{price(x5)} \\ \text{soar(e1)} \\ \text{agent(e1,x5)} \\ \text{patient(e1,x2)} \\ \text{record(x2)} \\ \text{level(x2)} \\ \hline \end{array} \qquad d(\mathrm{H}): \begin{array}{|c|}\hline \text{e1 x2 x3} \\ \hline \text{crude(x2)} \\ \text{oil(x3)} \\ \text{nn(x3,x2)} \\ \text{price(x2)} \\ \text{rise(e1)} \\ \text{agent(e1,x2)} \\ \hline \end{array}$$

Geographical Knowledge. Because a high number of examples in the development set required knowledge about geography, we automatically compiled a set of axioms from the CIA factbook (`http://www.cia.gov/cia/publications/factbook/` , covering knowledge about capitals, countries and US states. Some examples are:

$\forall x \forall y(\text{paris}(x) \wedge \text{france}(y) \rightarrow \text{in}(x,y))$
$\forall x \forall y(\text{pago\_pago}(x) \wedge \text{american\_samoa}(y) \rightarrow \text{in}(x,y))$

However, we could not find any examples where the theorem prover found a proof due to geographical knowledge in the test set.

3.5 Model Building

While theorem provers are designed to prove that a formula *is* a theorem (i.e., that the formula is true in any model), they are generally not good at deciding that a formula is *not* a theorem. Model builders are designed to show that a formula is true in at least one model. Hence, in addition to the Vampire theorem prover we also use the model builder Paradox [CS03].

To exploit the complementary approaches to inference, we use both a theorem prover and a model builder for any inference problem: the theorem prover attempts to prove the input whereas the model builder simultaneously tries to find a model for the negation of the input. If the model builder finds a model for

$$\neg(f(d(\mathrm{T})) \rightarrow f(d(\mathrm{H}))) \qquad (= \neg\mathrm{A})$$

we know that there cannot be a proof for its negation (hence no entailment). And if the model builder is able to generate a model for

$$f(d(\mathrm{T})) \wedge f(d(\mathrm{H})) \qquad (= \neg\mathrm{B})$$

we know that T and H are consistent (maybe entailment). (In practice, this is also a good way to terminate the search for proofs or models: if the theorem prover finds a proof for $\neg\phi$, we can halt the model builder to try and find a model for ϕ (because there won't be one), and vice versa.)

Another interesting property of a model builder is that it outputs a model for its input formula, if the input is satisfiable. A model is here the logical notion of a model, describing a situation in which the input formula is true. Formally, a model is a pair $\langle D, F\rangle$ where D is the set of entities in the domain, and F a function mapping predicate symbols to sets of domain members. For instance, the model returned for $f(d(\mathrm{T}))$ in Example 78 is one where the domain consists of three entities (domain size = 3):

```
D = {d1,d2,d3}       F(loc) = {}
F(book) = {d1,d2}    F(seller) = {}
F(clinton) = {d3}    F(be) = {}
F(of) = {(d1,d3)}    F(agent) = {}
F(big) = {}          F(patient) = {}
```

Model builders like Paradox generate such finite models by iteration. They attempt to create a model for domain size 1. If they fail, they increase the domain size and try again, until either they find a model or their resources run out. Thus, although there are possibly infinitely many models, model builders generally build a model with a minimal domain size. (For more information on model building consult [BB05]).

In the next section we show how to explore these finite models by using the domain and model size in predicting entailment.

3.6 Approximating Entailment

In an ideal world we calculate all the required background knowledge and by either finding a proof or a countermodel, decide how T and H relate with respect to entailment. However, it is extremely hard to acquire all the required background knowledge. This is partly due to the limitations of word sense disambiguation, the lack of resources like WordNet, and the lack of general knowledge in a form suitable for automated inference tasks.

To introduce an element of robustness into our approach, we use the models as produced by the model builder to measure the "distance" from an entailment. The intuition behind it is as follows. If H is entailed by T, the model for T+H is not informative compared to the one for T, and hence does not introduce new entities. Put differently, the domain size for T+H would equal the domain size of T. In contrast, if T does not entail H, H normally contains some new information (except when it contains negated information), and this will be reflected in the domain size of T+H, which then is larger than the domain size of T. It turns out that this difference between the domain sizes is a useful way of measuring the likelihood of entailment: large differences are mostly not entailments, small differences mostly are.

Consider the following example:

Example: 1049 (TRUE)

T: Four Venezuelan firefighters who were traveling to a training course in Texas were killed when their sport utility vehicle drifted onto the shoulder of a highway and struck a parked truck.

H: Four firefighters were killed in a car accident.

Although this example is judged as a true entailment, Vampire (the theorem prover that we use) does not find a proof because it lacks the background knowledge that one way of causing a car accident is to "drift onto the shoulder of the highway and strike something". On the other hand, Paradox, the model builder that we use, generates a model with domain size 11 for $f(d(\mathrm{T}))$, and a model with domain size 12 for $f(d(\mathrm{T}))\wedge f(d(\mathrm{H}))$. The absolute difference in domain sizes is small, and therefore likely to indicate an entailment. Apart from the absolute difference we also compute the difference relative to the domain size. For the example above the relative domain size yields $1/12 = 0.083$.

The domain size only tells us something about the number of entities used in a model—not about the number of established relations between the model's entities. Therefore, we also introduce the notion of model size. The model size is defined here by counting the number of all instances of two-place relations (and three-place relations, if there are any) in the model, and multiplying this with the domain size. For instance, the following (arbitrary) model

```
D = {d1,d2,d3}
F(cat) = {d1,d2}
F(john) = {d3}
F(of) = {(d1,d3)}
F(like) = {(d3,d1),(d3,d2)}
```

has a domain consisting of three entities and three instantiated two-place relations, yielding a model size of $3 * 3 = 9$.

Obviously, it is harder for model builders to generate a minimal model than just any model. In practice, a model builder like Paradox generally constructs models with a minimal domain size, but not necessarily one with a minimal model size. It is unclear how much this influenced our results but we plan to experiment with other model builders in future work, or taking aboard algorithms that transfer a model into a minimal model.

3.7 Deep Semantic Features

Given our approach to deep semantic analysis, we identified eight features relevant for recognising textual entailment. The theorem prover provides us with two features: `entailed` determining whether T implies H, and `inconsistent` determining whether T together with H is inconsistent. The model builder gives us six features: `domainsize` and `modelsize` for T+H as well as the absolute and relative difference between the sizes of T and T+H, both for the size of the domains (`domainsizeabsdif`, `domainsizereldif`) and the size of the models (`modelsizeabsdif`, `modelsizereldif`).

4 Experiments

There are not many test suites available for textual inference. We use throughout this section the dataset made available as part of the RTE challenge. We used the t-test for the difference between two proportions to measure whether the difference in accuracy between two algorithms or an algorithm and the baseline is statistically significant at the 5% level.

4.1 Dataset Design and Evaluation Measures

The organisers released a development set of 567 sentence pairs and a test set of 800 sentence pairs. In both sets, 50% of the sentence pairs were annotated as TRUE and 50% as FALSE, leading to a 50% most frequent class baseline for automatic systems.

The examples are further distinguished according to the way they were designed via a so-called *Task* variable. Examples marked CD (Comparable Documents) comprise sentences with high lexical overlap in comparable news articles, whereas the hypotheses of examples marked QA (Question Answering) were formed by translating questions from e.g., TREC into statements. The other subsets are IE (Information extraction), MT (Machine Translation) RC (Reading Comprehension), PP (Paraphrase Acquisition) and IR (Information Retrieval).

The different examples and subsets cover a wide variety of different aspects of entailment, from incorporation of background knowledge to lexical to syntactic entailment and combinations of all these. For a more exhaustive description of dataset design we refer the reader to [DGM05].

4.2 Experiment 1: Human Upper Bound

To establish a human upper bound as well as investigate the validity of the datasets issued, one of the authors annotated all 800 examples of the test set for entailment, following the short RTE annotation guidelines available at `http://www.pascal-network.org/Challenges/RTE/Instructions`. The annotation was performed before the release of the gold standard annotation for the test set and was therefore independent of the organisers' annotation. The organisers' and the author's annotation yielded a high percentage agreement of 95.25%. However, 33% of the originally created examples were already filtered out of the corpus before release by the organisers because of agreement-related problems. Therefore we expect that human agreement on textual entailment in general is rather lower. A further discussion of the gold standard dataset can be found in Section 6.

4.3 Decision Trees for Entailment Recognition

We expressed each example pair as a feature vector, using different subsets of the features described in Section 2 and Section 3 for each experiment. We then trained a decision tree for classification into TRUE and FALSE entailment on the development set, using the Weka machine learning tool [WF00], and tested on the test set.

Apart from a classification, Weka also computes a confidence value between 0.5 and 1 for each decision, dependent on the leaf in the tree that the classified example falls into: if the leaf covers x examples in the training set, of which y examples are classified wrongly, then the error rate is y/x and the confidence value is $1 - y/x$.

Following the RTE challenge, the evaluation measures are accuracy (*acc*) as the percentage of correct judgements as well as confidence-weighted average score (*cws*), which rewards the system's ability to assign a higher confidence score to correct judgements than wrong ones [DGM05]: after the n judgements are sorted in decreasing order by their confidence value, the following measure is computed:

$$cws = \frac{1}{n}\sum_{i=1}^{n}\frac{\#\text{correct-up-rank-}i}{i}$$

All evaluation measures are computed over the whole test set as well as on the 7 different subsets (CD, IE, etc.). The results are summarised in Table 1. We also computed precision, recall and F-measure for both classes TRUE and FALSE and will discuss the results in the text whenever of interest.

Experiment 2: Shallow Features. In this experiment only the shallow features (see Section 2) were used. The overall accuracy of 56.9% is significantly higher than the baseline.

Column 2 in Table 1 shows that this decent performance is entirely due to excellent performance on the CD subset. (Recall that the CD set was designed explicitly with examples with high lexical overlap in mind.)

In addition, the method overestimates the number of true entailments, achieving a Recall of 0.926 for the class TRUE, but a precision of only 0.547 on the same class. In contrast, it has good precision (0.761) but low recall (0.236) for the FALSE class. Thus, there is a correspondence between low word overlap and FALSE examples (see also the discussion of Example 731 in Section 2); high overlap, however, is normally necessary but not sufficient for TRUE entailment (see also Example 78 in Section 3).

Table 1. Summary of Results for Experiments 1 to 6

Exp	1: Human		2: Shallow		3: Strict		4: Deep		5: Hybrid		6: Hybr/Task	
Task	acc	cws	acc	cws	acc	cws	acc	cws	acc	cws	acc	cws
CD	0.967	n/a	**0.827**	**0.881**	0.547	0.617	0.713	0.787	0.700	0.790	**0.827**	0.827
IE	0.975	n/a	0.508	0.503	**0.542**	0.622	0.533	0.616	**0.542**	0.639	**0.542**	**0.627**
MT	0.900	n/a	0.500	0.515	0.500	0.436	**0.592**	**0.596**	0.525	0.512	0.533	0.581
QA	0.961	n/a	0.531	0.557	0.461	0.422	0.515	0.419	0.569	0.520	**0.577**	**0.531**
RC	0.979	n/a	0.507	0.502	**0.557**	0.638	0.457	0.537	0.507	0.587	**0.557**	**0.644**
PP	0.920	n/a	0.480	0.467	0.540	0.581	0.520	0.616	0.560	**0.667**	**0.580**	0.619
IR	0.922	n/a	0.511	0.561	0.489	0.421	0.567	0.503	**0.622**	**0.569**	0.611	0.561
all	0.951	n/a	0.569	0.624	0.520	0.548	0.562	0.608	0.577	0.632	**0.612**	**0.646**

Experiment 3: Strict Entailment. To test the potential of entailment as discovered by theorem proving alone, we now use only the `entailment` and `inconsistent` features. As expected, the decision tree shows that, if a proof for T implies H has been found, the example should be classified as TRUE, otherwise as FALSE. The `inconsistent` feature was not used by the decision tree, which was not surprising as very few examples were covered by that feature.

The deep semantic analysis was able to create semantic representations and then search for proofs for 774 of all 800 T/H-pairs in the test data, achieving a coverage of 96.8%. The precision (0.767) for the class TRUE is reasonably high:

if a proof is found, then an entailment is indeed very likely. The exceptions where we found a proof but entailment did not hold are discussed further in Section 5. However, recall is very low (0.058) as only 30 proofs were found on the test set (for some examples see Section 3). This yields an F-measure of only 0.10 for the TRUE class. Due to the low recall, the overall accuracy of the system (0.52, see Table 1) is not significantly higher than the baseline.

Thus, this feature behaves in the opposite way to shallow lexical overlap and overgenerates the FALSE class. Missing lexical and background knowledge is the major cause for missing proofs.

Experiment 4: Approximating Entailment. As discussed in Section 3.6 we now try to compensate for missing knowledge and improve recall for TRUE entailments by approximating entailment with the features that are furnished by the model builder. Thus, Experiment 4 uses all eight deep semantic analysis features, including the features capturing differences in domain- and modelsizes. The recall for the TRUE class indeed jumps to 0.735. Although, unavoidably, the FALSE class suffers, the resulting overall accuracy (0.562, see Column 4 in Table 1) is significantly higher than when using the features provided by the theorem prover alone (as in Experiment 3). The confidence weighted score also rises substantially from 0.548 to 0.608. The approximation achieved can be seen in the different treatment of Example 1049 (see Section 3.6) in Experiments 3 and 4. In Experiment 3, this example is wrongly classified as FALSE as no proof can be found; in Experiment 4, it is correctly classified as TRUE due to the small difference between domain- and modelsizes for T and T+H.

There is hardly any overall difference in accuracy between the shallow and the deep classifier. However, it seems that the shallow classifier in its current form has very little potential outside of the CD subset whereas the deep classifier shows a more promising performance for several subsets.

Experiment 5: Hybrid Classification. As shallow and deep classifiers seem to perform differently on differently designed datasets, we hypothesized that a combination of these classifiers should bring further improvement. Experiment 5 therefore used all shallow and deep features together. However, the overall performance of this classifier (see Column 5 in Table 1) is not significantly better than either of the separate classifiers. Closer inspection of the results reveals that, in comparison to the shallow classifier, the hybrid classifier performs better or equally on all subsets but CD. In comparison to the deep classifier in Column 4, the hybrid classifier performs equally well or better on all subsets apart from MT. Overall, this means more robust performance of the hybrid classifier over differently designed datasets and therefore more independence from dataset design.

Experiment 6: Dependency on Dataset Design. As Experiment 5 shows, simple combination of methods, while maybe more robust, will not necessarily raise overall performance if the system does not know when to apply which method. To test this hypothesis further we integrated the subset indicator as a feature with the values CD, IE, MT, RC, IR, PP, QA into our hybrid system.

Indeed, the resulting overall accuracy (0.612) is significantly better than either shallow or deep system alone. In addition, the performance over all subsets is now more even and robust.

Note that using both a combination of methodologies *and* the subset indicator is necessary to improve on individual shallow and deep classifiers for this corpus. We integrated the subset indicator also into the shallow and deep classifier by themselves, yielding classifiers Shallow+Task and Deep+Task, with no or only very small changes in accuracy (these figures are not included in Table 1).

5 Error Analysis

A full error analysis of the hybrid system is beyond the scope of this paper as it incorporates a multitude of factors, including errors of shallow and deep methods as well as errors induced by the learning model and the combination of methods. We will, however, discuss error types of the shallow word overlap and the main theorem proving component of our system in the following two subsections.

5.1 Shallow Methods

As discussed in Experiment 2 above, the word overlap method tends to overgenerate the TRUE class. Typical examples that lead to such false positives can be summarized as follows:

– Negation that is not present in the hypothesis but is present in the text (see Example 78 in Section 3).
– Structural conversions, as we use a bag-of-words model only (for example, active-passive conversions).
– Conditional information in text or hypothesis, for example in “if”-clauses; ordinals; idioms. This includes Examples 1617, 2040, 2025, 2055, 2030, 2082 and 2079, which were also a problem for the theorem proving component and are in detail discussed in Section 5.2 below.
– Underestimation of crucial non-matching words in the hypothesis (see Example 828 below, where the mismatch of *20-year-old* is underestimated due to considerable overlap of other words like *Jennifer Hawkins*, *Australia*, *beauty queen* etc.).

  Example: 828 (FALSE)

  T: Jennifer Hawkins is the 21-year-old beauty queen from Australia.

  H: Jennifer Hawkins is Australia’s 20-year-old beauty queen.

– Word sense ambiguity. As the shallow method does not perform any WSD, this can lead to incorrectly relating lemmata in hypothesis and text like *hit* and *shot* in Example 1959 below.

  Example: 1959 (FALSE)

  T: Kerry hit Bush hard on his conduct on the war in Iraq.

  H: Kerry shot Bush.

5.2 Strict Entailment

Experiment 3 showed that strict entailment creates many false negatives due to missing background knowledge. In contrast, when the theorem prover found a proof it was usually correct. Only 30 proofs were found by the system, of which 23 were annotated as entailments in the gold standard. These include adequately analysed phenomena such as apposition (5 times: 760, 929, 995, 1903, 1905), relative clauses (3 times: 142, 1060, 1900), coordination and attachment(3 times: 898, 807, 893), active-passive alternation (twice: 1007, 1897), possessives (once: 1010), the use of background knowledge (6 times: 236, 836, 1944, 1952, 1987, 1994) and more or less straightforward cases (3 times: 833, 1076, 741). Two of such examples (1005 from the training set and 898 from the test set) are given in Section 3.

Incorrect proofs were found for seven cases. It is interesting to find out why our system discovered a proof in these cases. It turns out that they are due to incorrect lexical semantics, the lack of dealing with metaphors, the restricted expressivity of first-order logic, and the inability to deal with idiomatic expressions. We will discuss these cases in detail.

Ordinals were not dealt with correctly in Examples 1617 and 2040. In both cases the relative clause (1617) and the infinitive construction (2040) were not part of the restriction of the ordinal, giving an incorrect semantic representation:

Example: 1617 (FALSE)

T: In 1782 Martin Van Buren, the first US president who was a native citizen of the United States, was born in Kinderhook, N.Y.

H: The first US president was born in Kinderhook, N.Y.

Example: 2040 (FALSE)

T: Stjepan Mesic was the first Croatian president to deliver a public address at Harvard.

H: Stjepan Mesic was the first Croatian president.

Example 2025 shows a text with a conditional. The current system does not adequately deal with all discourse adverbials yet, causing it to assert that Poland joins the EU rather than placing it in the antecedent of a conditional.

Example: 2025 (FALSE)

T: There are a lot of farmers in Poland who worry about their future if Poland joins the European Union.

H: Poland joins the European Union.

A rather similar case is Example 2055, where the system correctly associated Einstein to be the subject of being the president of Israel, but it incorrectly assumed that being invited to X is being X. A restriction on this class of modal verbs will fix this problem. (In the development data, however, there were similar cases that were annotated as entailments.)

Example: 2055 (FALSE)

T: The fact that Einstein was invited to be the president of Israel is critical to an accurate understanding of one of the greatest individuals in modern history.

H: Einstein is the president of Israel.

An interesting example is 2030. Here it seems that *capital* in 2030-T is used metaphorically, in the sense that it is the most important location with respect to gastronomy. As our system did not spot this, it incorrectly found a proof instead.

Example: 2030 (FALSE)

T: Lyon is actually the gastronomic capital of France.

H: Lyon is the capital of France.

A neo-Davidsonian analysis based on first-order logic is problematic for cases such as Example 2082. Although we analyse the modifiers of a verb all intersectively, it seems that *established in Italy* should be analysed restrictively.

Example: 2082 (FALSE)

T: Microsoft was established in Italy in 1985.

H: Microsoft was established in 1985.

Example 2079, finally, shows that some entailment pairs require a sophisticated analysis of idiomatic expressions:

Example: 2079 (FALSE)

T: US presence puts Qatar in a delicate spot.

H: Qatar is located in a delicate spot.

These examples of false positives show once more how hard recognising textual entailment is. Overall, however, the backbone of our deep semantic analysis is reasonably accurate. Its recall for TRUE entailments can be increased by finding methods for selecting appropriate background knowledge, and revising some of the lexical semantics will improve its precision.

6 Discussion of the Entailment Task

We will now discuss some observations we made on the task definition and the annotated data sets.

6.1 Task Definition

The current RTE dataset classified entailment as binary TRUE and FALSE. Following FRACAS, the semantic test suite in [CCVE^{+}96], a classification that respects three values (yes, don't know, inconsistent), is probably more in its place. For instance, not only are examples 1301 and 1310 below not entailments, the hypotheses are inconsistent with the corresponding texts as well:

Example: 1301 (FALSE) (sic)

T: The former wife of the South African president did not ask for amnesty, and her activities were not listed in the political reports submitted by the African National Congress to the Truth and Reconciliation Commission in 1996 and 1997.

H: Winny Mandela, the President's ex-wife, is requesting amnesty.

Example: 1310 (FALSE) (sic)

T: Although the hospital insists that King Hussein is not fully free of the cancer, they are hopeful that he will recover.

H: The statement added that King Hussein has been cured completely.

6.2 Annotated Datasets

When establishing the upper bound in Experiment 1, we made several observations that could have an effect on the design of future test suites. The disagreements in annotation fell roughly into three categories. Firstly, the amount of background knowledge assumed by our annotation sometimes differed from the one taken into account by the gold standard annotation. Thus, the annotating author judged Example 825 below as FALSE, as she was not aware that Beiji was in the *north* of Iraq. The question of how much background knowledge can be assumed and how many examples should draw on extensive background knowledge in future datasets was extensively discussed at the RTE workshop.

Example: 825 (TRUE) (own annotation FALSE)

T: A car bomb that exploded outside a U.S. military base near Beiji, killed 11 Iraqis.

H: A car bomb exploded outside a U.S. base in the northern town of Beiji, killing 11 Iraqis.

Secondly, in a few cases, sentences that were isolated from their original context in the examples made it hard to annotate them correctly. Thus, our own annotation for Example 961 was FALSE as the first name of *Seiler* as *Audrey* is not inferrable from the text without a wider context.

Example: 961 TRUE (own annotation FALSE)

T: Seiler was reported missing March 27 and was found four days later in a marsh near her campus apartment.

H: Abducted Audrey Seiler found four days after missing.

Thirdly, several entailments were incorrectly annotated the gold standard in our opinion. Example 236 (see below), for instance, was judged as entailment. But taking tense into account (which, incidentally, our system is currently not able to do), it is strictly speaking not a textual entailment.

Example: 236 (TRUE) (sic)

T: Yasir Arafat has agreed to appoint a longtime loyalist as interior minister to take charge of the country's security.

H: Yasir Arafat nominated a loyalist as interior minister.

Another example is 893: the adverb *perhaps* in the text clearly expresses doubt on the date of establishment of settlements on Jakarta, and the hypothesis establishes it as a fact. This clearly is not entailment.

Example: 893 (TRUE) (sic)

T: The first settlements on the site of Jakarta were established at the mouth of the Ciliwung, perhaps as early as the 5th century AD.

H: The first settlements on the site of Jakarta were established as early as the 5th century AD.

7 Related Work

Our shallow analysis is similar to several shallow models presented as part of the RTE challenge, in particular the word overlap methods by [JdR05] and the shallow baselines by [HPV05]. The results achieved by us and them give a current upper bound of 55–57% accuracy for shallow overlap methods. This is also confirmed by [DGM05], who cite a non-participating system by Rada Mihalcea based on shallow features only with an accuracy of 56.8%. [PA05] adapted the BLEU algorithm that relies mainly on n-gram overlap. They report that the resulting system did not beat the 50% baseline. This difference in results might be due to the fact that the BLEU algorithm is not directional, i.e. it penalises difference in information and length between text and hypothesis also if the hypothesis is completely included and entailed by the text but, for example, contains much *less* information. All these approaches confirm the fact that shallow methods do best on the CD subtask and have the tendency to overestimate TRUE entailments. IDF models for entailment and question answering, but not within the RTE framework, have also been proposed by [MdR03, SGH$^+$04].

The main idea of our deep analysis, using a detailed semantic analysis and first-order inference, goes back to [BB05]. Other approaches to textual entailment that are comparable to our "strict" entailment as carried out in Experiment 3 include the OTTER theorem prover [Akh05, FHH$^+$05] and EPILOG in [BBF$^+$05]. Both these systems (like our "strict" system) do not beat the baseline, with [BBF$^+$05] explicitly mentioning lack of background knowledge and inference rules as the reason, confirming our experience. We incorporate model building as a central part of the inference mechanism as a partial solution to this problem, an approach not adopted by any other system as far as we know. We have shown that using model generation is a promising way to approximate entailment and can improve the low recall of theorem proving. It is interesting to compare our approach to [RNM05]. Although they do not use model builders, a similar basic idea of relaxing the constraints of strict theorem proving and weighing the differences between hypothesis and text underlies their weighted abduction approach.

As far as we are aware, our combination of a shallow with a deep approach into a single hybrid system is unique and adds to the robustness of our approach. We also show that differences in dataset design can be exploited by such a hybrid approach, resulting in a significant overall improvement. The task variable was also explored by [NSDC05] and [RNM05] with neither of them reporting statistically significant differences between their systems with and without the task variable. We believe the reason for this discrepancy compared to our results might lie in the fact that their systems do not incorporate two fundamentally different inference strategies. As discussed in Section 4, using the task variable with the deep or shallow approach alone does not improve results, whereas the combination of two different strategies plus the task variable yields improvements.

Results of other approaches to determining textual entailment indicate that it is an extremely hard task. The RTE workshop revealed that participating systems reached accuracy figures ranging between 0.50 and 0.59 and cws scores between 0.50 and 0.69 [DGM05]. Comparing this with our own results (accuracy 0.61 and cws 0.65) shows how well our systems performs on the same data set.

8 Conclusions

Relying on theorem proving as a technique for determining textual entailment yielded high precision but low recall due to a general lack of appropriate background knowledge. We used model building as an innovative technique to surmount this problem to a certain extent. Still, it will be unavoidable to incorporate automatic methods for knowledge acquisition to increase the performance of our approach. Future work will be directed to the acquisition of targeted paraphrases (as for example in [BL03]) that can be converted into background knowledge in the form of axioms.

Our hybrid approach combines shallow analysis with both theorem proving and model building and achieves high accuracy scores on the RTE dataset compared to other systems that we are aware of. The results for this approach also indicate that (a) the choice of entailment recognition methods might have to vary according to the dataset design and/or application and (b) that a method that wants to achieve robust performance across different datasets might need the integration of several different entailment recognition methods as well as an indicator of design methodology or application.

Acknowledgements

We would like to thank Mirella Lapata and Malvina Nissim for their comments on earlier versions of this paper. We are also grateful to Valentin Jijkoun and Bonnie Webber for discussion on textual entailment and Steve Clark and James Curran for help on using the CCG-parser. Thanks to James for producing the fancy CCG-derivations in Section 3.

References

[Akh05] E. Akhmatova. Textual entailment resolution via atomic propositions. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, pages 61–68, 2005.

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

[BBF$^+$05] S. Bayer, J. Burger, L. Ferro, J. Henderson, and A. Yeh. Mitre's submission to the eu pascal rte challenge. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, pages 41–44, 2005.

[BBKdN01] Patrick Blackburn, Johan Bos, Michael Kohlhase, and Hans de Nivelle. Inference and Computational Semantics. In Harry Bunt, Reinhard Muskens, and Elias Thijsse, editors, *Computing Meaning Vol.2*, pages 11–28. Kluwer, 2001.

[BCS$^+$04] J. Bos, S. Clark, M. Steedman, J. Curran, and J. Hockenmaier. Wide-coverage semantic representations from a ccg parser. In *Proc of the 20th International Conference on Computational Linguistics; Geneva, Switzerland; 2004*, 2004.

[BL03] Regina Barzilay and Lillian Lee. Learning to paraphrase: An unsupervised approach using multiple sequence alignment. In *NAACL-HLT 2003*, 2003.

[Bos03] J. Bos. Implementing the Binding and Accommodation Theory for Anaphora Resolution and Presupposition Projection. *Computational Linguistics*, 2003.

[Bos05] Johan Bos. Towards wide-coverage semantic interpretation. In *Proceedings of Sixth International Workshop on Computational Semantics IWCS-6*, pages 42–53, 2005.

[CC04] S. Clark and J.R. Curran. Parsing the WSJ using CCG and Log-Linear Models. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL '04)*, Barcelona, Spain, 2004.

[CCVE$^+$96] R. Cooper, R. Crouch, J. Van Eijck, C. Fox, J. Van Genabith, J. Jaspars, H. Kamp, M. Pinkal, D. Milward, M. Poesio, and S. Pulman. Using the framework. fracas: A framework for computationla semantics. Technical report, Fracas Deliverable D16., 1996.

[CS03] K. Claessen and N. Sörensson. New techniques that improve mace-style model finding. In *Model Computationa - Principles, Algorithms, Applications (Cade-19 Workshop)*, Miami, Florida., 2003.

[DGM05] Ido Dagan, Oren Glickman, and Bernardo Magnini. The pascal recognising textual entailment challenge. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, pages 1–8, 2005.

[Fel98] Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database.* MIT Press, Cambridge, Mass., 1998.

[FHH$^+$05] A. Fowler, B. Hauser, D. Hodges, I. Niles, A. Novischi, and J. Stephan. Applying cogex to recognize textual entailment. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, pages 69–72, 2005.

[HPV05] Jesus Herrera, Anslemo Penas, and Felisa Verdejo. Textual entailment recognition based on dependency analysis and wordnet. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, 2005.

[JdR05] Valentin Jijkoun and Maarten de Rijke. Recognising textual entailment using lexical similarity. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, 2005.

[KR93] Hans Kamp and Uwe Reyle. *From Discourse to Logic. Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory.* Kluwer, Dordrecht, Netherlands, 1993.

[MdR03] C. Monz and M. de Rijke. Light-weight entailment checking for computational semantics. In *Proc. of the 3rd Workshop on Inference in Computational Semantics; 2003*, 2003.

[MS99] Christopher Manning and Hinrich Schuetze. *Foundations of Statistical Natural Language Processing.* MIT Press, 1999.

[NSDC05] Eamonn Newman, Nicola Stokes, John Dunnion, and Joe Carthy. Ucd iirg approach to the textual entailment challenge. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, 2005.

[PA05] Diana Perez and Enrique Alfonseca. Application of the bleu algorithm for recognising textual entailments. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, 2005.

[RNM05] R. Raina, A.Y. Ng, and C. Manning. Robust textual inference via learning and abductive reasoning. In *Proc. of AAAI 2005*, 2005.

[RV02] A. Riazanov and A. Voronkov. The design and implementation of Vampire. *AI Communications*, 15(2-3), 2002.

[SGH$^+$04] H. Saggion, R. Gaizauskas, M. Hepple, I. Roberts, and M Greenwood. Exploring the performance of boolean retrieval strategies for open domain question answering. In *Proc. of the Information Retrieval for Question Answering (IR4QA) Workshop at SIGIR 2004*, 2004.

[Ste01] M. Steedman. *The Syntactic Process.* The MIT Press, 2001.

[VdS92] R.A. Van der Sandt. Presupposition Projection as Anaphora Resolution. *Journal of Semantics*, 9:333–377, 1992.

[WF00] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann, San Diego, CA, 2000.