When logical inference helps determining textual entailment (and when it doesn't)

Johan Bos
Dipartimento di Informatica
Università di Roma "La Sapienza"
via Salaria 113
00198 Roma
bos@di.uniroma1.it

Katja Markert
School of Computing
University of Leeds
Woodhouse Lane
Leeds, LS2 9JT
markert@comp.leeds.ac.uk

Abstract

We compare and combine two methods to approach the second textual entailment challenge (RTE-2): a shallow method based mainly on word-overlap and a method based on logical inference, using first-order theorem proving and model building techniques. We use a machine learning technique to combine features of both methods. We submitted two runs, one using only the shallow features, yielding an accuracy of 61.6%, and one using features of both methods, performing with an accuracy score of 60.6%. These figures suggest that logical inference didn't help much. Closer inspection of the results revealed that only for some of the subtasks logical inference played a significant role in performance. We try to explain the reason for these results.

1 Introduction

In this paper we summarise the results of our efforts in the 2005/2006 Recognising Textual Entailment (RTE-2) challenge (the task of deciding, given two text fragments, whether the meaning of one text is entailed/inferred from another text). We experimented with shallow and deep semantic analysis methods. The shallow techniques were used to establish a baseline performance, but also to complement the deep semantic analysis. Both methods extract features from the training data, which are then combined using an off-the-shelf machine learning classifier. We will introduce the shallow semantic analysis (Section 2) and the deep semantic analysis (Section 3), present the results of our two runs (Section 4), and discuss them (Section 5).

Our work for RTE-2 is essentially a further development of our approach to RTE-1 (Bos and Markert, 2005). Changes in the deep analysis (see Section 3) constitute a refinement of semantic analysis, the computation of background knowledge, and the use of more advanced model builders. In addition, we also refined the method for ranking the output, which is used for calculating average precision.

2 Shallow Semantic Analysis

As observed by several researchers, including ourselves, the RTE-1 dataset (Ido Dagan and Magnini, 2005) showed a remarkably frequent dependency between surface string similarity of text and hypothesis and the existence of entailment. Therefore, high word overlap is likely to be an important feature for determining textual entailment (see, for example, T/H pair 217 in RTE-2) whereas the occurrence of words in the hypothesis that are unrelated to any words in the text make entailment unlikely.

We use a bag-of-words model to measure word overlap `wnoverlap`, where the weights of lemmas in the hypothesis that are related to lemmas in the text are added to the overlap measure and other, unrelated lemmas are ignored. Weights correspond to inverse document frequency with the Web as corpus and relatedness includes equality as well as synonymy and morphological derivations in WordNet. For a more detailed description we refer the reader

to (Bos and Markert, 2005). In addition, we use four other shallow features: `textlength` measuring the length of T in words, `hyplength` measuring the length of the hypothesis and `proplength` measuring the difference between `textlength` and `hyplength` as $\frac{textlength}{hyplength}$. The last shallow feature `task` simply uses the `task` variable (one of `SUM, QA, IE, IR`) as our results in RTE-1 (Bos and Markert, 2005) showed that the different tasks can need different inference methods.

3 Deep Semantic Analysis

3.1 Semantic Representation

We use a robust wide-coverage CCG-parser (Bos, 2005) to generate fine-grained semantic representations for each T/H-pair. The semantic representation language is a first-order fragment of the DRS-language used in Discourse Representation Theory (Kamp and Reyle, 1993), conveying argument structure with a neo-Davidsonian analysis and including the recursive DRS structure to cover negation, disjunction, and implication. Third-person personal pronouns are resolved to named entities, and proper names and definite descriptions are treated as anaphoric too. They are bound to previously introduced discourse referents if possible, otherwise accommodated.

For more details of our approach the reader is kindly referred to (Bos and Markert, 2005). The appendix at the end of this paper contains some example representations.

3.2 Inference

To check whether an entailment holds or not, we used two kinds of automated reasoning tools: first-order theorem proving, and finite model building. We used the standard translation from DRS to first-order logic (Kamp and Reyle, 1993) to map the semantic representations onto the format required by the inference tools. We employed the theorem prover Vampire 7 (Riazanov and Voronkov, 2002) and two model builders, Paradox 1.3 (Claessen and Sörensson, 2003) and Mace 2.0 (McCune, 1998).

For each T-H pair we calculated possible relevant background knowledge (BK) (see Section 3.3 for a description of the background knowledge used). We then ran the following inferences:

1. proof $T \rightarrow H$ (Vampire)
2. proof $(BK \wedge T) \rightarrow H$ (Vampire)
3. proof $\neg(BK \wedge T)$ (Vampire)
4. proof $\neg(BK \wedge T \wedge H)$ (Vampire)
5. satisfy $BK \wedge T$ (Paradox, Mace)
6. satisfy $BK \wedge T \wedge H$ (Paradox, Mace)

In 1 and 2 we are directly checking for a logical entailment between T and H; in 1 we try so without, and in 2 with background knowledge. (Note that if there is a proof found for 1, there will always be a proof for 2).

In 3 and 4 we check for consistency of the background knowledge with T or H. Sometimes the combination of background knowledge with the text or hypothesis causes a logical inconsistency. This can be due to errors in the syntax-semantics interface, or to errors in the generation of relevant background knowledge. In both cases, if Vampire is able to find a proof, we know that we are dealing with inconsistent background knowledge. (Note that if there is a proof for 3, it logically follows that there is a proof for 4 as well.)

In 5 and 6 we generate first-order models for T and H with supporting background knowledge. This is only possible if there are no proofs found for 3 and 4. We perform model building using two different model builders: We use Paradox to find the size of the domain, and then use Mace to construct a minimal model giving that domain size. (The reason for this is efficiency: Paradox is generally faster than Mace, but doesn't always produce minimal models. Mace generally produces minimal models.)

The results of 1–6 give us a set of features that can be used to determine whether it is likely that T entails H or not. If Vampire finds a proof for 1, it is very likely that T entails H. If Vampire finds a proof for 3 or 4, then we are (unfortunately) dealing with inconsistent background knowledge and there is nothing we can say about the relationship between T and H (however, it could be the case that T + H is inconsistent, which would mean that T does not entail H). Else, if there is no proof for 3 or 4, but there is a proof for 2, then again it is very likely that T entails H. Finally, if the model size difference between

the models generated for 5 and 6 is very small then it is likely that T entails H.

3.3 Background Knowledge

We generate background knowledge (BK) using two kinds of sources: hyponymy relations from WordNet, and a set of manually coded inference rules expressing general knowledge (see Appendix for examples). Lexical knowledge is created automatically from WordNet. A hyponymy relation between two synsets A and B is converted into $\forall$x(A(x)$\rightarrow$B(x)). Two synset sisters A and B are translated into $\forall$x(A(x)$\rightarrow \neg$B(x)).

Rules for generic knowledge cover the semantics of possessives, active-passive alternation, spatial knowledge, causes of death, winning prizes or awards, family relations, diseases, producers, employment, and ownership. Some examples of such rules are given in the appendix.

There are 115 of these rules, constructed manually based on the development data. Although this way of manual construction is not easily scalable the extracted relations can be used for automatic bootstrapping of further relations in future work.

3.4 Model Building

An attractive property of a model builder (such as Mace or Paradox) is that it outputs a model for its input formula (only of course if the input is satisfiable). A model is here the logical notion of a model, describing a possible 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.

Model builders like Paradox and Mace generate 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. This way the models they output are generally minimal models; in other words, the models do not contain entities that are not mentioned in the input (see also (Blackburn and Bos, 2005)).

To introduce an element of robustness into our logical inference approach, we use the models as produced by the model builders 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 introduce 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: RTE-2 504 (YES)

T: Never before had ski racing, a sport dominated by monosyllabic mountain men, seen the likes of Alberto Tomba, the flamboyant Bolognese flatlander who at 21 captured two gold medals at the Calgary Olympics.

H: Alberto Tomba won a ski race.

Although this example is judged as a true entailment, Vampire doesn't find a proof because it lacks the background knowledge that capturing gold medals means that you must have won a race. Vampire generated a model with domain size 15 for T, and a model with domain size 16 for T plus 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 $1/16 = 0.0625$.

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 relations in the model, and multiplying this with the domain size.

3.5 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 four features: `entailed` and `entailedBK` determining whether T implies H (without or with background knowledge), and `inconsistentT` and `inconsistentTH` determining whether T or

Table 1: Summary of Results

Exp	Run 1		Run 2	
Task	prec	av. prec.	prec	av. prec
IE	0.505	0.462	**0.550**	**0.550**
IR	**0.660**	**0.713**	0.640	0.723
QA	**0.565**	**0.606**	0.530	0.598
SUM	**0.735**	**0.814**	0.705	0.746
all	**0.616**	**0.669**	0.606	0.604

T/H are inconsistent with the background knowledge. The model builder gives us four features: 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 and Results

We submitted two runs, both ranked. The results on the test set are summarised in Table 1.

4.1 Run 1: Only shallow features

Our first run used only the shallow features described in 2. We used the machine learning tool WEKA (Witten and Frank, 2000) to derive a decision tree model from the development data:

```
wnoverlap <= 0.771437: NO (373.0/130.0)
wnoverlap > 0.771437
|  task = IR: YES (60.0/18.0)
|  task = IE
|  |  textlength <= 41: YES (125.0/48.0)
|  |  textlength > 41: NO (20.0/3.0)
|  task = QA
|  |  textlength <= 34
|  |  |  wnoverlap <= 0.95583: NO (51.0/22.0)
|  |  |  wnoverlap > 0.95583: YES (86.0/24.0)
|  |  textlength > 34: NO (18.0/4.0)
|  task = SUM: YES (67.0/7.0)
```

WEKA computes a *confidence value* per leaf: For example, on the SUM task the YES value covers 67 examples, of which only 7 are classified wrongly. WEKA therefore computes an error rate of 7/67 for this leaf and a confidence value of $1 - 7/67$. We used the confidence value as the primary ranking criterion for our decisions. As the confidence value is the same within an individual leaf we used features with numeric values for secondary ranking, prioritising features which were used for earlier splitting; thus, we used `wnoverlap` for secondary sorting and `textlength` for tertiary sorting. Thus, the most confident YES decisions are SUM YES-decision with high overlap and low textlength.

The model achieved 68% precision on the whole development data, 64% precision on the development set using ten-fold cross-validation and 61.6% precision on the test set. The fall in precision can be explained by the fact that the average word overlap in the test set is higher (overlap median in development set is 0.79 vs. 0.84 in the test set); rescaling of the `wnoverlap` variable via linear functions might alleviate this problem. The method achieved good results in average precision (66.9% over the whole test set), indicating that sorting by decreasing overlap is a useful ranking criterion.

4.2 Run 2: Combining shallow and deep

For a combination of deep and shallow features we took into account that (a) some example pairs could not be handled by the deep semantic analysis, and (b) that examples for which the theorem prover found a proof make an entailment highly likely. (When using more features we found that the task feature led to overfitting on the development data; therefore the task feature was left out of all Run 2 experiments.) Therefore, we split the test data into three subsets: $Test_{shallow}$ contains examples which could not be handled via the deep analysis because the examples either could not be parsed or because the theorem prover discovered an inconsistency of T or T/H with the background knowledge. $Test_{entailed}$ contains consistent and parsed examples where Vampire found a proof, either with or without background knowledge and $Test_{combined}$ contains all remaining examples. The same subsets exist for the development data. For decisions on $Test_{shallow}$, which contains 112 examples, a decision tree using the shallow features only was trained on $Dev_{shallow}$ (containing 57 examples). The precision on this subset alone was only 56.5%, possibly due to the small training set or to the fact that this subset contains some of the most complex examples.

$Test_{entailed}$ contains 29 examples, of which 19 proofs were found without background knowledge and 10 with background knowledge Proofs were mostly correct, with 22 out of 29 proofs being correct (precision of 76%), being therefore the single most reliable piece of evidence in our system, but having low recall. Phenomena that the

deep semantic approach was able to handle include monotone inclusion appositions incorporation of world knowledge, conjunctions, pronoun resolution, active-passive conversion and relative clauses. All examples in $Test_{entailed}$ were assigned a YES value, using the accuracy on the development set as confidence value.

On $Test_{combined}$ a decision tree using shallow and non-shallow features was trained on $Dev_{combined}$. The resulting tree

```
domainsizereldif <= 0.285714
|  wnoverlap <= 0.771437: NO (260.0/108.0)
|  wnoverlap > 0.771437
|  |  textlength <= 41: YES (316.0/108.0)
|  |  textlength > 41: NO (24.0/4.0)
domainsizereldif > 0.285714: NO (106.0/22.0)
```

emphasises differences in domain-sizes and shallow overlap and achieves 65.7% on the development set and 60.2% on the test set.

The overall accuracy of the combined methods on the whole test set was 60.6%. The overall combined ranking over the whole test set used the confidence values of all three subsets as primary sorting constraint with differences in domain size as secondary and word overlap as tertiary sorting criterium.

5 Discussion

The first and the second run perform with almost the same overall precision, which is disappointing as one would expect the deep analysis to enhance the shallow one. We found that there are the following main reasons for this lack of improvement: firstly, the recall of the best deep feature (entailment) is quite low and actually mostly finds proofs for examples that also have a high word overlap (for example, most examples of monotone inclusion will also have a word overlap of 1). Similarly, small domain size differences correlate with high overlap, although domain size differences also handle multi-word entities and can draw on background knowledge.

Overall, this means that the two systems have a high degree of overlap in their decisions. The ranking of Run 2 performs *worse* than the one for Run 1. This is mainly due to the combination of confidence values from different decision trees, which might not be straightforward to compare. However, Run 2 is more robust across the different subtasks, achieving results that are better than the 50% baseline on all subsets, whereas Run 1 does not beat the baseline for the IE task.

References

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

Johan Bos and Katja Markert. 2005. Recognising textual entailment with logical inference. In *Proc. of the 2005 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 628–635.

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

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

Oren Glickman Ido Dagan and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*.

H. Kamp and U. Reyle. 1993. *From Discourse to Logic; An Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and DRT*. Kluwer, Dordrecht.

W. McCune. 1998. Automatic Proofs and Counterexamples for Some Ortholattice Identities. *Information Processing Letters*, 65(6):285–291.

A. Riazanov and A. Voronkov. 2002. The Design and Implementation of Vampire. *AI Communications*, 15(2–3).

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

Appendix: Example Analysis

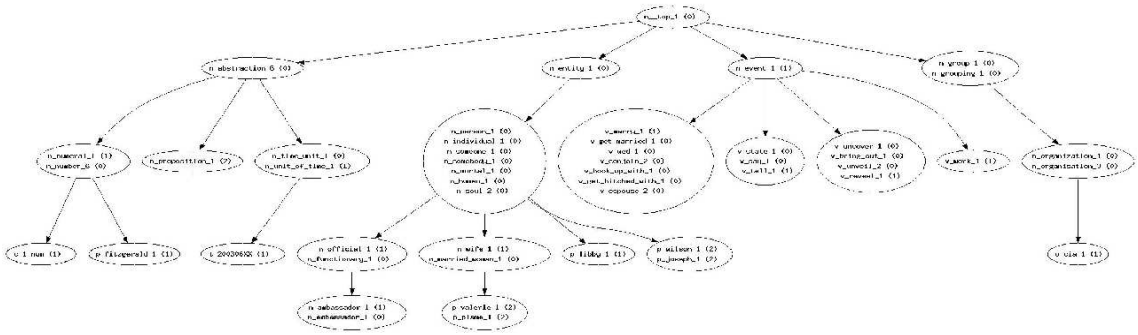
Example: RTE-2 98 (YES)

T: Mr. Fitzgerald revealed he was one of several top officials who told Mr. Libby in June 2003 that Valerie Plame, wife of the former ambassador Joseph Wilson, worked for the CIA.

H: Valerie Plame is married to Joseph Wilson.

DRS for **H**:

```
 _____________________
| x3 x1 x2            |
|---------------------|
| named(x3,plame,per) |
| named(x3,valerie,per) |
| marry(x1)           |
| patient(x1,x3)      |
| named(x2,wilson,per) |
| named(x2,joseph,per) |
| to(x1,x2)           |
|_____________________|
```



DRS for **T**:

```
 ____________________________________
| x4 x16 x12 x17 x15 x11 x1 x2       |
|------------------------------------|
| male(x4)                           |
| named(x4,fitzgerald,per)           |
| named(x4,mr,ttl)                   |
| named(x16,libby,per)               |
| named(x16,mr,ttl)                  |
| named(x12,plame,per)               |
| named(x12,valerie,per)             |
| named(x15,joseph,per)              |
| nn(x17,x15)                        |
| ambassador(x17)                    |
| named(x15,wilson,per)              |
| former(x15)                        |
| named(x11,cia,org)                 |
| reveal(x1)                         |
| agent(x1,x4)                       |
| theme(x1,x2)                       |
|     ______________________________ |
|    | x5 x6 x7 x9 x8 x3           | |
|  x2:|-----------------------------| |
|    | |x5|>1                      | |
|    | top(x6)                     | |
|    | official(x6)                | |
|    | timex(x8)=2003-06-XX        | |
|    | tell(x7)                    | |
|    | patient(x7,x16)             | |
|    | agent(x7,x6)                | |
|    | theme(x7,x9)                | |
|    |      ___________________    | |
|    |     | x14 x13 x10       |   | |
|    |  x9:|-------------------|   | |
|    |     | wife(x14)         |   | |
|    |     | of(x14,x15)       |   | |
|    |     |      ___________  |   | |
|    |     |     |           | |   | |
|    |     |  x13:|-----------| |   | |
|    |     |      | x12 = x14 | |   | |
|    |     |      |___________| |   | |
|    |     | work(x10)         |   | |
|    |     | agent(x10,x12)    |   | |
|    |     | for(x10,x11)      |   | |
|    |     |___________________|   | |
|    | in(x7,x8)                   | |
|    | of(x5,x6)                   | |
|    |      _________              | |
|    |     |         |             | |
|    |  x3:|---------|             | |
|    |     | x4 = x5 |             | |
|    |     |_________|             | |
|    |_____________________________| |
|____________________________________|
```

Background Knowledge (BK):

```
 _____________      _____________
| x18 x16 x17 |    |             |
|-------------|    |-------------|
| in(x18,x16) | ==> | in(x18,x17) |
| in(x16,x17) |    |_____________|
|_____________|

 _____________      _____________
| x13 x15 x14 |    |             |
|-------------|    |-------------|
| event(x13)  | ==> | in(x15,x14) |
| agent(x13,x15) |  |_____________|
| in(x13,x14) |
|_____________|
 _____________      _____________
| x10 x12 x11 |    |             |
|-------------|    |-------------|
| event(x10)  | ==> | in(x12,x11) |
| patient(x10,x12) |  |_____________|
| in(x10,x11) |
|_____________|

 _____________      _____________
| x7 x9 x8    |    |             |
|-------------|    |-------------|
| event(x7)   | ==> | in(x9,x8) |
| theme(x7,x9) |   |_____________|
| in(x7,x8)   |
|_____________|

 _____________      _____________
| x4 x5       |    | x6          |
|-------------|    |-------------|
| wife(x4)    | ==> | marry(x6)  |
| of(x4,x5)   |    | patient(x6,x4) |
|_____________|    | to(x6,x5)   |
                   |_____________|
 _____________      _____________
| x1 x2       |    | x3          |
|-------------|    |-------------|
| wife(x1)    | ==> | marry(x3)  |
| of(x1,x2)   |    | patient(x3,x2) |
|_____________|    | to(x3,x1)   |
                   |_____________|
```

Model for BK & T & H:

```
D=[d1, d2, d3, d4, d5, d6, d7, d8, d9, d10]
  f(1, n_abstraction_6, [d1, d7, d8])
  f(1, n_group_1, [d5])
  f(1, n_entity_1, [d2, d3])
  f(1, n_event_1, [d4, d6, d9, d10])
  f(1, n_numeral_1, [d1])
  f(1, n_proposition_1, [d7])
  f(1, v_reveal_1, [d6])
  f(1, v_tell_1, [d9])
  f(1, v_marry_1, [d4])
  f(1, n_unit_of_time_1, [d8])
  f(1, n_official_1, [d3])
  f(1, n_wife_1, [d2])
  f(1, v_work_1, [d10])
  f(1, p_wilson_1, [d3])
  f(1, p_libby_1, [d2])
  f(1, n_organization_1, [d5])
  f(1, n_person_1, [d2, d3])
  f(1, n_ambassador_1, [d3])
  f(1, t_200306XX, [d8])
  f(1, p_valerie_1, [d2])
  f(1, p_fitzgerald_1, [d1])
  f(1, o_cia_1, [d5])
  f(1, c_1_num, [d1])
  f(1, a_male_1, [d1])
  f(1, p_mr_1, [d1, d2])
  f(1, p_plame_1, [d2])
  f(1, p_joseph_1, [d3])
  f(1, a_former_1, [d3])
  f(1, a_top_1, [d3])
  f(2, card, [ (d1, d1)])
  f(2, r_in_1, [ (d2, d8), (d3, d8), (d7, d8), (d9, d8)])
  f(2, r_agent_1, [ (d6, d1), (d9, d3), (d10, d2)])
  f(2, r_patient_1, [ (d4, d2), (d4, d3), (d9, d2)])
  f(2, r_theme_1, [ (d6, d7), (d9, d7)])
  f(2, r_of_1, [ (d1, d3), (d2, d3)])
  f(2, r_to_1, [ (d4, d2), (d4, d3)])
  f(2, r_nn_1, [ (d3, d3)])
  f(2, r_for_1, [ (d10, d5)])
```