Textual Entailment Resolution via Atomic Propositions

Elena Akhmatova
Centre for Language Technology
Division of Information and Communication Sciences
Macquarie University
North Ryde, NSW, 2109
elena@ics.mq.edu.au

Abstract

This paper presents an approach to solving the problem of textual entailment recognition and describes the computer application built to demonstrate the performance of the proposed approach. The method presented here is based on syntax-driven semantic analysis and uses the notion of atomic proposition as its main element for entailment recognition. The idea is to find the entailment relation in the sentence pairs by comparing the atomic propositions contained in the text and hypothesis sentences.

The comparison of atomic propositions is performed via an automated deduction system OTTER; the propositions are extracted from the output of the Link Parser; and semantic knowledge is taken from the WordNet database. On its current stage the system is capable to recognize basic semantically and syntactically based entailments and is potentially capable to use more external and internal knowledge to deal with more complex entailments.

1 Introduction

The variety of ways to transmit the same information is an interesting phenomenon of natural language and is an obstacle for many applications in the domain of natural language processing. Question answering, for example, has faced the fact that a possible answer to a question could be expressed in a way that is syntactically and semantically different from the question sentence, or has to be entailed from it. The paper is devoted to the phenomena of entailment. By textual entailment is understood a relationship between a coherent text T and a language expression H, which is considered as a hypothesis. T entails H if the meaning of H, as interpreted in the context of T, can be deduced from the meaning of T. By a *language expression* is understood a syntactically coherent text fragment, having a well formed fully connected syntactic analysis (Dagan and Glickman, 2004). For example,

T: Coffee boosts energy and provides health benefits.
H: Coffee gives health benefits.

is a true textual entailment that will be used as an example throughout the paper.

2 Meaning Representation

To know if a hypothesis H is entailed from a text T one should compare their meanings. We represent meaning of a sentence as a set of atomic propositions contained in it and compare the propositions in order to compare the sentences. We mean by an *atomic proposition* a minimal declarative statement (or a small idea) that is either true (T) or false (F) and whose truth or falsity does not depend on the truth or falsity of any other proposition. (*Coffee boosts energy and provides health benefits.* – propositions are: *Coffee boosts energy.* and *Coffee provides health benefits*.)

To break a sentence into its atomic propositions a syntax-driven semantic analysis of the sentence (Jurafsky and Martin, 2000) is applied, as we believe that a deep semantic and syntactical analysis is vital to solve the problem.

The implementation of the method uses an output of the parser as an input for the semantic analyser producing the output from which a first-order logic representation of the meaning can be derived.

```
    +--------------------------------------------- Xp ---------------------------------------------+
    +------ Wd ----+---- Ss ----+---- Os ----+                                                      |
    |              |            |            |                                                      |
LEFT-WALL coffee.n boosts.v energy.n and provides.v health.n benefits.n.
```

⇒ [coffee]subj [provides] [health benefits]obj

```
exists x exists y exists z exists za6 (Subj(x) &iq(x, 'coffee') &Pred(y)&iq(y,
'provides') & Obj(z)&iq(z, 'benefits')&attr(z, za6) & Obj(za6)& iq(za6, 'health')).
```

```
    +--------------------------------------------- Xp ---------------------------------------------+
    |                                                         +------------- Op ----------+         |
    +------ Wd ----+---------------------- Ss ----------------------+          +---- AN ---+        |
    |              |                                                |          |           |        |
LEFT-WALL coffee.n boosts.v energy.n and provides.v health.n benefits.n.
```

⇒ [coffee]subj [boosts] [energy]obj

```
exists x exists y exists z (Subj(x) &iq(x, 'coffee') & Pred(y) &iq(y, 'boosts')
& Obj(z)&iq(z, 'energy') ).
```

…
↓
linkage array
↓
propositions
↓
logic form
↓
…

Figure 1. Data of the system on different stages of analysis.

This final meaning representation is called *the logic formula* of the sentence. See figure 1 for an example of the data the system has.

A logical meaning representation of a sentence and an automatic deduction system to work with it are often used in QA applications (Moldovan et al., 2002; Moldovan et al., 2003). There exist many approaches to describe meaning by means of a logical form. Thus, a sentence *A restaurant serves meat* can have a description

exists e, x Isa(e, Serving) & Server(e, x) & Served(e, Meat) & Isa(x, Restuarant) (Jurafsky and Martin, 2000).

From our point of view, these forms are rigid and hard to produce. There are no automatic rules to understand that the event here is *serving*, and the subject of the sentence is a *server*. It's not clear how to compare two such logical representations.

As a result we use a simplified representation in this version of the system. There are three types of objects - *Subj(x), Obj(x), Pred(x)* and a meaning attaching element *iq(x, <meaning of x>)*. See fig. 1 for an example. Also, there are two variants of relationships *attr(x, y)* and *prep(x, y):*

“Somali capital” -- *Subj(x) & iq(x, ‘capital’) & attr(x, y) & Subj(y) & iq(y, ‘somali’).*

“a zoo in Berlin” - *Obj(x) & iq(x, ‘zoo’) & prep(x, y) & Obj(y) & iq(y, ‘Berlin’).*

In this case a logic formula is easy to build automatically. Semantic synonymy is expressed as an equivalence (*iq*(*x*, ‘serve’) <-> *iq*(*x*, ‘dish’)) and hyperonymy ((*iq*(*x*, ‘serve’) -> *iq*(*x*, ‘provide’)), (*iq*(*x*, ‘serve’) -> *iq*(*x*, ‘cater’))). We can store lexical relation rules, *all x (iq(x, 'is') <-> iq(x, 'be'));* describe syntactical equivalence by means of additional logic rules, “Be X of Y -> X Y” (*director of the firm -> direct a firm*) is *all y z z1 y77 z77 ((Pred(z1) & iq(z1, 'be') & Obj(y) & iq(y, y77) & prep(y, z) & Obj(z) & iq(z, z77)) -> (Pred(y) & iq(y, y77) & Obj(z) & iq(z, z77)));* or soften some mistakes of the parser, such as prepositional attachment - *all x y y77 z z77 x77 (Pred(y) & iq(y, y77) & prep(y, x) & Obj(x) & iq(x, x77) & Obj(z) & iq(z, z77) <-> Pred(y) & iq(y, y77) & Obj(z) & iq(z, z77) & attr(z, x) & Obj(x) & iq(x, x77)).*

The rules are called *knowledge rules*, as they represent knowledge of the system.

3 WordNet Relatedness

A WordNet (WordNet) relatedness algorithm used in the system was developed specially for this system, as the existing ones (Budanitsky and Hirst, 2001) are not quite right for the system. The result of its work is a relatedness score. It is used to prove the synonymy or entailment relation between words (see figure 3 for details). As it compares senses of the words, a WSD algorithm could be used (will be in future) prior to the comparison to get a more reliable score (otherwise the probability that the current word has sense *i* could be estimated as 1/*n*, where *n* – number of senses the word has). The score is calculated from the paths between the senses of the words in the graph. We use the length of a path (*take over–buy* has a length 2, *form-make-establish* has a length 3; the longer the path, the less is the relatedness); the amount of sense of the words that is on the path between these two words (two words connected with a verb *print*, for example, are more close to each other, as the words connected via *make*, because *print* has only 4 senses and make has 49); and the total number of different paths (the words which senses are connected through 10 different paths are more related then the words having only one connecting path, for example). Though we compare all words similarly now, I would like to emphasize that the following method ideally should be used only for verbs and nouns derived from them, and a different one in other cases, for, intuitively, the verbs (think about *decide* and *conclude*, and the nouns *decision* and *conclusion* derived from them) have a more generic meaning, than the nouns describing particular objects (*train*, *car*, *bus*).

4 System description

The scheme of the system is presented on the figure 2. The Link Parser 4.1a (Link Grammar) to trace the connections between the elements of sentences and a version 3.3 of OTTER (OTTER) for comparison of the atomic propositions are used now. A way to logic form is shown in section 2. After the algorithm is the following (figure 2: Otter and its input data; and figure 4): if for every proposition in the hypothesis sentence there is one in the text sentence from that it could be entailed then the sentence entailment holds, otherwise the entailment does not hold.
The same algorithm also can be used to obtain some knowledge rules from the data set: when entailment holds we want to find pairs <p1, p2> (see fig. 2) and to build knowledge rules p1->p2 to use them later. The idea of this process is the following: for every atomic proposition Y in the hypothesis find the atomic proposition in the text from which it is entailed. If there is none, find the closest (with the higher relatedness score according to WordNet (see section 7)) atomic proposition X and create a rule Prop X -> Prop Y. So, what we can learn is:
reduce the risk of diseases **->** *have health benefits.*

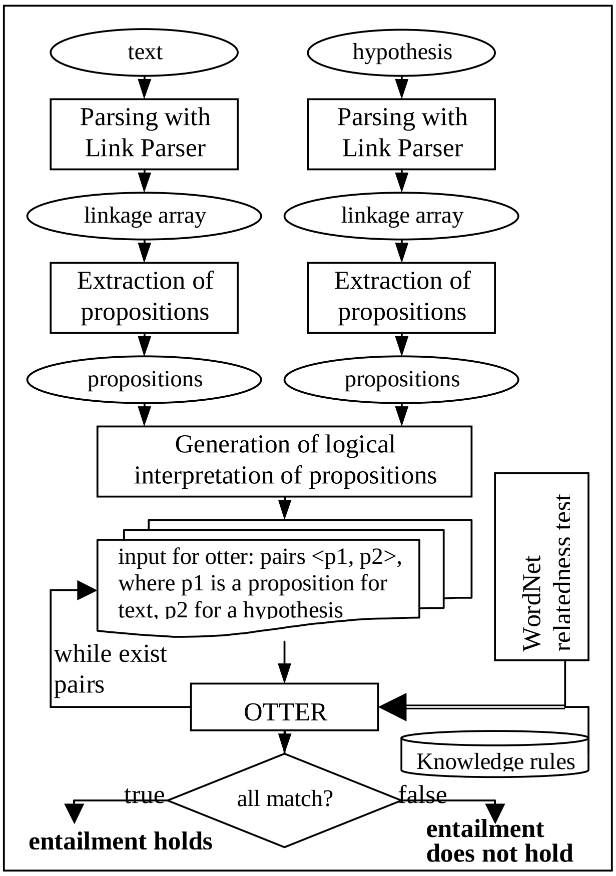

Figure 2. System architecture.

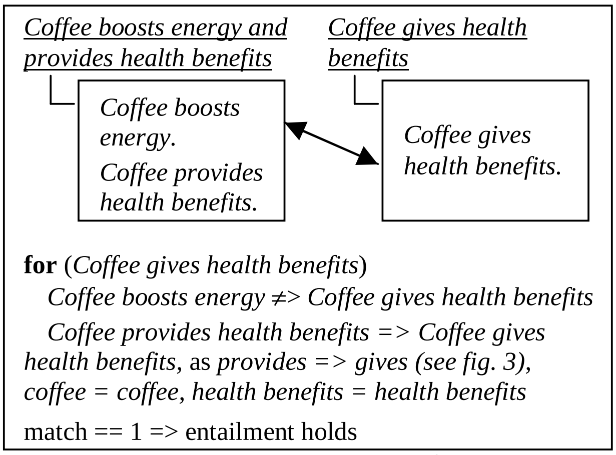

Figure 4. Comparison of propositions.

provide – give **relatedness score: 0,3623**
(verb chain); maximum path length – 3 => ∀ x (*iq*(*x*, 'provide') -> *iq*(*x*, 'give')).

1. **provide#1(7)**[2259805] -- *hyperonym* -- **give#3(44)**[2136207]
…
7. **provide#6(7)**[2155855] -- *hyperonym* -- **support#2(11)**[2155507] -- *hyperonym* -- **give#3(44)**[2136207]

Figure 3. WordNet relatedness algorithm. Data and results.

5 Performance of the system

First, the examples where entailment holds and it is right.

T: *The decision is made.* - H: *The determination is made.*

As *decision* and *determination* are connected via WordNet, so we'll have a rule *all x (iq(x, 'decision') <-> iq(x, 'determination'))* .

T: *The good decision is made.* - H: *The decision is made.*

Subj(x) & iq(x,'decision') & attr(x, y) & Subj(y) & iq(y, 'good') -> Subj(x) & iq(x,'decision')

T: *The Brazilian president visited France.* - H: *The president of Brazil visited France,* and T: *The boy goes to school by bus.* - H: *The boy travels with school bus.*

A rule *all x y (attr(x, y) <-> prep(x, y))* works here.

T: *The man is a director of the company.* – H: *The man rules the company.*

"Be X of Y -> X Y" (section 2) rule is used here.

Now, the examples where entailment holds though it shouldn't:

T: *The population of France has grown during the last 3 years.* – H: *The population of Paris has grown during the last 3 years*.

T: *The gastronomic capital of France is Lyon.* – H: *The capital of France is Lyon.*

T: *The man came to the park by car.* – H: *The man came to a car park.*

It is clear now why the following two examples were recognized as TRUE entailments:

T: *A male gorilla escaped from his cage in the Berlin zoo and sent terrified visitors running for cover, the zoo said yesterday.* – H: *A gorilla escaped from his cage in a zoo in Germany.*

T: *The incident in Mogadishu, the Somali capital, came as U.S. forces began the final phase of their promised March 31 pullout.* – H: *The capital of Somalia is Mogadishu.*

6 Results

cws: 0.5067; accuracy: 0.5188 ;
precision: 0.6119; recall: 0.1025; f: 0.1756

task	cws	accuracy	task	cws	accuracy
CD	0.6121	0.5867	RC	0.4702	0.5214
IE	0.5519	0.5083	PP	0.5452	0.5200
MT	0.4341	0.4917	IR	0.4797	0.5111
QA	0.4649	0.4769			

Note: according to *Recognising Textual Entailment Challenge* evaluation method (Pascal Challenges).

The results are low now, as more work should be done for proposition extraction and logical representation. Also a good knowledge rule database is missing.

7 Future work

Despite not very high results we believe the proposed system has a strong potential. The main future tasks are: to make inferences inside the text sentence itself, to try reasoning with all propositions from the text, and to create an inference rule database. An attempt will be done to construct the database using sentences with inferences inside them. That is the sentences with the conjunctions *as result of, because, if,* and predicates *cause, follow*.

References

Alexander Budanitsky and Graeme Hirst. 2001. *Semantic Distance in WordNet: An Experimental, application-oriented evaluation of five measures.* Proceedings of the NAACL−2001 Workshop on WordNet and Other Lexical Resources, Pittsburg, PA.

Cordell Green. 1969. *Theorem-Proving by Resolution as a Basis for Question-Answering Systems*. Machine Intelligence, Chapter 11. Edinburgh University Press, pp. 183-205.

Dan Moldovan et al. 2002. *LCC Tools for Question Answering*. NIST Special Publication 200 – 251.

Dan Moldovan et al. 2003. COGEX: A Logic Prover for Question Answering. Proc. HLT-NAACL 2003, Edmonton, 2003.

Daniel Jurafsky and James H. Martin. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition.* Prentice-Hall. Chapter 14-15.

Ido Dagan and Oren Glickman. 2004. Probabilistic Textual Entailment: *Generic Applied Modeling of Language Variability. Learning Methods for Text Understanding and Mining,* 26 - 29 January 2004, Grenoble, France.

Link Grammar. http://www.link.cs.cmu.edu/link/.

OTTER. www-unix.mcs.anl.gov/AR/otter.

Pascal Challenges. *Recognising Textual Entailment Challenge*. http://www.pascal-network.org.

WordNet. www.cogsci.princeton.edu/~wn.