Harald Hammarstrom 770804-6912
Uppsala Master's Thesis in Computing Science 220
Examensarbete DV3
2002-08-28
ISSN 1100-1836

Shallow Reasoning on Open Domain Texts

Supervisor: Maarten de Rijke, ILLC, Univ. of Amsterdam, mdr@science.uva.nl
Examiner: Roland Bol, Dept. of IT, Uppsala Univ., rolandb@csd.uu.se

Harald Hammarstrom
haha2581@student.uu.se
Information Technology
Computing Science Department
Uppsala University
Box 337
S-751 05 Uppsala
Sweden

Abstract

This thesis addresses methods to estimate if a natural language text contains (at least) all the information contained in another text (i.e *entailment*). A simple method is based on the analysis of word frequencies (idf-word statistics), disregarding their context. Other methods compare larger chunks of text (verb groups, prepositional phrases and noun phrases). We aim to enhance the idf-based method with ideas from from Natural Language Processing and Information Retrieval, notably anaphora resolution and taking synonyms into account. To compare the methods, we compare their judgments with a human judgment, that we assume to be perfect. The comparison combines precision and recall. We can conclude that idf-word statistics is by far the best considered option to estimate the distribution of semantic content. Thus the near future lies in giving the idf-based method a larger text base to work on rather than switching away from word-level analysis.

Contents

Acknowledgments

First and foremost many thanks go to my supervisors Maarten de Rijke and Roland Bol who were kind enough to make my project possible. Also this thesis wouldn't have come through without practical help from Ingrid van Loon and technical help from Christof Monz. Last but not least my fellow undergraduate Magnus Ågren has provided moral support and invaluable help on LaTeXcommands as well as allowance to run heavy experiments on his supercomputer.

Chapter 1

Introduction

The aim of this project is to devise, implement and test methods for computing semantic entailment relations between text passages. This comprises a natural subtask in various areas of Natural Language Processing (NLP) and as such this kind of tasks have mostly been done by generating first-order logic representations of natural language documents which are then fed into a first-order theorem prover. This is an ideal plan to solve such problems, but at the present state of knowledge it has certain limitations. First, efficient and sufficiently robust Natural Language Processing tools are not available. Secondly, the first-order theorem provers themselves might require vast computational costs. Also, as logic inference tools they stick to strict yes/no answers which is not always desirable.

With this background, we look at variants of lighter representations with a lower computational cost level. Starting from a quite minimal version, we investigate how it can be exploited before switching to a richer representation and the extensions and limitations of that. Like everywhere in Natural Language Processing and Information Retrieval (IR), it's hard to know a priori the exact balance between robustness and computational cost. To which extent do deeper levels of analysis help? And for which tasks, if any, do they help? Answering these questions satisfactorily calls uncompromisingly for a *functioning evaluation procedure*.

The entailment task can be measured meaningfully by using the standard Information Retrieval notions of precision and recall, applied to a set of entailment problems with a human-made key.

What we call 'lighter representation' is the bag-of-words model, where we see the text as simply its words lumped together *entirely stripped of their*

relative order in the text. With 'richer representation' we use a partial parser[1] to divide the text into chunks of mainly *Noun Phrases* NPs[2], *Prepositional Phrases* PPs and *Verb Groups* VGs. These are kept with order within their sentence, whereas the full sentences are then bagged without order.

It seems that richer representations should produce equally or more accurate results at the cost of more computation. One goal of this work is to prove or disprove this conjecture, another is to characterize the trade-off, if any, thereof. Moreover, we wish to identify the limit of reasoning with either representation, i.e after having applied all available NLP tools that are worth applying, what's the maximum performance? What does this maximum performance mean in absolute terms? Apart from the value in knowledge about this, what kind of practical use can it provide?

[1]Parsing is described on p.13 in chapter 2.3.1 Parsing

[2]The unacquainted reader can check Appendix A.1 on p.80 for a quick intro on some linguistic terminology used in this thesis

Chapter 2

Background

2.1 What is an entailment relation?

If what is said in one segment of natural language text is included in what is said in another piece of text, we say that the latter *entails* the former. The entailing segment of text is presumably at least as large in content as the entailed one and could contain extra information - that's not important. The key is that one piece of text is *at least as informative* as the other. Thus, as a constructed example we can give that

"Jane loves dogs and cats."

entails

"Jane loves cats."

But the latter does not entail the former. Another constructed illustration that requires a little more inference is

"Jane loves dogs but hates all other four-legged animals."

entails

"Jane hates cats."

but not vice versa. A trained linguist would also note here that we do not care about subtleties such as focus.

To be more precise, we are comparing the semantic content of segments of natural language text (at this stage the particular language English). One should start to worry right here about how to actually interpret this formally because, as is obvious, natural language does not come with any hopes of strict distinctions[1].

Also, to decide whether A entails B (i.e. implies) in the general case is undecidable - one could easily express problems like the halting problem in natural language. And decidable entailment relations could come to require exponential time to compute. Even if we imagine entailment relations on the universe of text segments to form a nice (non-total) partial order, we cannot expect to fully sort this out - neither theoretically nor in practice.

So we lower the goals and make the following two important assumptions: Let humans be judges of the uncertainties inherent in normal natural language and have them discern if A entails B or not. And trust this judgement, called the *gold standard*.

Aside from NL related hardships, difficult entailment relations occur very scarcely in natural language texts (they are more numerous in textbooks of logic and such, but there mostly not formulated in the meta-language) and if they do we can afford a few mistakes.

For a sample of entailment relations together with the their human assessments actually used in this project, see the sections on the test set 2.4.1 beginning p.19.

2.2 What is the use of entailment relations?

Presumably, a great deal of natural language processing questions could be reformulated as entailment propositions (not necessarily strict yes/no propositions), if we had had access to perfect NLP tools. Even very early on in Information Retrieval, van Rijsbergen [van Rijsbergen, 1986] defined 'aboutness' of a document d in terms of logical entailment, where d 'is about' q if $d \models q$.

However, without any translation work, need for computing entailment relations arise naturally for example in the NLP sub-field Informativity. For

[1] Appendix A.2 on p.82 has a little more elaboration of this point

instance, when processing through a discourse, Informativity is to understand how the next information item relates to the current generated discourse context. It could be entailed in the context already, it could be new i.e. *informative* or at least partly so. [Blackburn et al., 1999] provide the theoretical framework to do this in terms of DRT *Discourse Representation Theory* as developed in [Kamp and Reyle, 1993].

[Gardent and Webber, 2000] show how semantic entailment relations can help disambiguate certain types of noun-noun compounds. More generally, some kind of entailment wizard could rule out unwanted models generated by a *model builder*. A model builder for NLP takes a set of natural language sentences and returns models (models exactly as in normal predicate logic) satisfying those sentences. If the set of sentences is inconsistent then obviously no models can be generated, so in addition you get a (non-prefect of course) check for inconsistency and satisfiability. Usually however, there are many satisfying models and the task is to single out the 'best suited' one(s) i.e. those that include all the information in the sentences and no extra unwarranted information. This is done by choosing the 'minimal' one, but it would be a great advantage to have entailment relations to exclude possible models. At today's stage of computing semantic entailment relations this would at best be to give preference to various models, not to give advice with certainty.

For browsing and retrieving information in large sets of documents, it is of great help to be able to arrange into categories and/or to summarize many overlapping documents. For instance, in the case below where we have nicely paragraphed news stories about the same event, a summarizer could merge them - taking care only to add subsequent paragraphs that are not already entailed.

[Barzilay et al., 1999] do summarizing essentially by locating two or more sentences that are very similar. Then some of these are considered superfluous and can be removed. To compute similarity between two sentences, they use an algorithm that compares the predicate-argument structure constituents generated by a statistical part-of-speech parser. As shall be seen, this is the same approach used in the second part of this project, chapter **4 Richer Representation Reasoning** on p.56. The advantage, in this case, by going via entailment relations is that work on entailment relations is easier to evaluate. Barzilay et al. complain that they had not found a collection of human written summaries that could serve as a gold standard. Moreover, since putting all the full content sentences together is done in their case

by some non-trivial text regeneration, vast additional complications would result even if they had a gold standard to match to.

[Radev, 2000] introduces CST *Cross Document Structure Theory* - a much more detailed analysis of inter-document relations. He lists 24 inter-document relationships such as equivalence, contradiction, historical background, follow-up, judgment, parallel, translation and more. A journalist user wanting a summary can then, say, choose to include background, judgments and follow-ups but no equivalents or translations of a certain document. If we can compute entailment relations, we can cut down considerably on the work required to produce this taxonomy by hand.

Some Question/Answering systems[2] such as that of [Harabagiu et al., 1999] have an intermediate stage, or a stage for justification after generating the answer, where certain paragraphs are highlighted to show where the answer is thought to lie. It's conceivable that an entailment engine can serve to select or rank which paragraphs are more or less likely to contain the answer. But this is only really useful if the Q/A engine does not use the same techniques as the entailment engine.

Perhaps the most suitable application for entailment relations, given that we cannot currently compute them with great accuracy, is **Topic Detection and Tracking** (TDT). TDT is about discovering and threading together topically related text material, thus it's somewhat related to document summarizing and document categorization. The full extent and organization of TDT research is given at http://www.nist.gov/TDT[3]. That text A entails text B obviously implies that the topic of B is in the topic of A, in fact, as shall be discussed in chapter **5** on Conclusions, there is basis for arguing that the methods examined in this thesis show *topic* entailment rather than content entailment.

2.3 How are entailment relations computed?

2.3.1 Background on NL understanding

Two steps are mandatory to be passed in order to compute entailment between Natural Language text segments, namely:

[2]Question/Answer systems are what they sound like - they take a natural language question and try to find the answer in a large NL text corpus

[3]Accessed the 22nd of March 2002 [TDT, 2002]

- To generate a suitable representation of the NL texts
- To do inference with this representation

The most exact representation of a natural language text would be its corresponding formalization in first-order (or for the sake of completeness, nth-order) predicate logic. But as can be expected, state-of-the-art tools for this on open domains are not even close to achieving such kind of exactness. This is not because research is effortless or faulty - progress is made continually - but because natural language understanding has shown itself to be extremely complex[Allen, 1995].

For instance, any textbook about natural language and logic will have example sentences of ambiguities on various levels, anomalies, presuppositions, conventional implicatures etc. inherent in the richness of natural language. Grasping the meaning i.e. generating a semantic representation of such phenomena as contexts grow above single clear example sentences, are the really challenging tasks for a computer.

But there's no sense in giving up, there are things we can do still. For this we will now introduce *tagging* and *partial parsing* that generate intermediate level understanding of NL text, which may or may not suffice (this is an empirical question) depending on the task.

Tagging

Tagging is the first part of going through a natural language text. The computer runs through the text, looks each word (or punctuation mark etc) up in a dictionary. For each word it assigns a tag of part-of-speech information, which is mainly the word-class of the word *as it's used in each particular case*, and also gives the word's *lemma* i.e the word stem stripped of morphological inflections.

Tagging can be done quite robustly; even on open domains the best implementations tag 96%-97% of the words correctly [Manning and Schütze, 2000]. Today's taggers all use some kind of statistical likelihood approach to disambiguate words that, depending on the context, can belong to several word classes, so the difference between different taggers is mainly their performance and how detailed a tag set they use. Standardized and named tagsets vary between say 30 to roughly 200 tags[4], thus they give a little more detail

[4] [Manning and Schütze, 2000] p. 140

than just word class. Fig 2.1 has a list of the most frequently used tags for English[5]:

Tag	Part of Speech	Example
AT	article	the, a
BEZ	the word *is*	is
IN	preposition	in
JJ	adjective	big
JJR	comparative adjective	bigger
MD	modal	may
NN	singular or mass noun	water, car
NNP	singular proper noun	Ireland
NNS	plural noun	Nigerians, cars
PERIOD	strong punctuation mark	. : ? !
PN	personal pronoun	I, me
RB	adverb	strongly
RBR	comparative adverb	higher[a]
TO	the word *to*	to
VB	verb, base form	say
VBD	verb, past tense	stole
VBG	verb, present participle/gerund	saying
VBN	verb, past participle	stolen
VBP	verb, non 3rd person sing. pres.	have
VBZ	verb, 3rd person sing. pres.	has
WDT	wh-determiner	what, which, who

Figure 2.1: Some frequently used part-of-speech tags for English

[a] as in "he jumped *higher*"

As can be seen, the tags classes are chosen according to a mixture of two criteria. According to how easy it is to computationally distinguish certain classes in English, as in the difference between say adjectives in the positive and comparative, and how much use there actually is of some distinction, as for example, we could have had more easily recognized distinctions on the pronoun level if it was common that people wanted this information.

For the experiments in this thesis, we have used TreeTagger [Schmid, 1994],

[5] [Manning and Schütze, 2000] p. 342, examples are mine

a decision-tree-based part-of-speech tagger. The exact tagset of which is given in **Appendix A.6** on p. 90. For more information on taggers we refer to chapter 4 and 10 of [Manning and Schütze, 2000], but figure 2.2 has a real text example with its tagged output.

Criticism of the tag assignments in this particular case can come on two levels.

- First, as we noted that taggers don't come with 100% accuracy and with say even 95%, which admittedly sounds very high and safe, means we can expect roughly one error per sentence in a regular newspaper article (since the average such has about 20 words). This is of course a valid argument, but we can afford it.

- From the theoretical linguist's point of view, some of the tag assignments are arbitrary or simply wrong. For example, is *party* as in '*party* battle' above to be tagged a noun when it functions like an adjective? It's no error according to the tagger to label *Protestant* in '*Protestant* party' an adjective, but what about its capital p? Linguistic hair-splitters like these are however not the point. Taggers are not trying to define well-founded and well-categorized parts of speech, they are used to preprocess text for further analyzing. And in doing so for a wide variety of purposes, it's better that they are conservative and cautious in their categorizing. Deeper analysis can then be done ontop with minimal information distortion.

Parsing

Parsing as in parsing grammar refers in general to splitting a sentence into its main syntactical constituents. Traditionally this means identifying subject, object, predicate, instrument, time and more. Because of syntactical ambiguity this is remarkably difficult. Consider for instance

> "He shot the cat in pyjamas"

As we humans understand it, 'in pyjamas' must refer to *he*, but it's not straightforward for a computer how to treat prepositional phrases like this. Its placement does not seem to be crucial because

- The syntactically analogous "he shot the man in pyjamas" we take 'in pyjamas' to refer to *the man* or

BELFAST, Northern Ireland (AP) The leader of Northern Ireland's biggest Protestant party narrowly won a crucial party battle Saturday, keeping alive the province's power-sharing government but only by promising to punish his coalition partners.

Word	Lemma	Tag	Tag Explanation
BELFAST	NNP	BELFAST	proper noun
,	,	,	
Northern	NNP	Northern	proper noun
Ireland	NNP	Ireland	proper noun
BELFAST	NNP	BELFAST	proper noun
(	(	(	
AP	NNP	AP	proper noun
)	)	)	
The	DT	the	determiner
leader	NN	leader	sing. noun
of	IN	of	preposition
Northern	NNP	Northern	proper noun
Ireland	NNP	Ireland	proper noun
's	POS	's	possessive ending
biggest	JJS	big	adjective superlative
Protestant	JJ	Protestant	adjective (positive)
party	NN	party	sing. noun
narrowly	RB	narrowly	adverb
won	VBD	win	verb past tense
a	DT	a	determiner
crucial	JJ	crucial	adjective
party	NN	party	sing. noun
battle	NN	battle	sing. noun
Saturday	NNP	Saturday	proper noun
,	,	,	
keeping	VBG	keep	verb gerund
alive	JJ	alive	adjective
the	DT	the	
province	NN	province	sing. noun
's	POS	's	possessive ending
power-sharing	JJ	power-sharing	adjective
government	NN	government	sing. noun
but	CC	but	conjunction
only	RB	only	adverb
by	IN	by	preposition
promising	VBG	promise	verb gerund
to	TO	to	the word *to*
punish	VB	punish	verb base form
his	PRP\$	his	possesive pronoun
coalition	NN	coalition	sing. noun
partners	NNS	partner	plural noun
.	SENT	.	sentence border

Figure 2.2: Some tagged output

- 'in pyjamas' could refer to the whole action taking place in a giant pair of pyjamas, analogous to "he shot the cat in the hotel room".

It's our knowledge of cats usually not wearing pyjamas and that shootings don't take place inside garments that enables us to make unambiguous sense of the sentence. In short, computers today cannot do the same.

What we can do however, quite robustly, is partial parsing (also known as chunk parsing). Chunk parsing identifies Noun Phrases (NPs), Verb Groups (VGs) and Prepositional Phrases (PPs) but it does not try to attach PPs or identify which NP is the subject etc. Exactly how chunk parsers achieve this is a whole science of its own, for which we recommend chapter 12 of [Manning and Schütze, 2000] or the less statistically focused [Jurafsky and Martin], but it's safe to say that the most popular ones rely on assigning different probabilities to different parses from which they select the maximal.

Due to mostly practical considerations, in this project we use PaPa, a statistical chunk parser implemented by Christof Monz[6]. Figure 2.3 has some sample PaPa-produced information. In addition, PaPa gives further knowledge about the phrase such as semantic head and possible dependencies (where applicable) which is quite valuable. The semantic head is the phrase constituent that carries the main lexical meaning, the other being qualifiers/modifiers to it. With dependencies we mean which chunks that *possibly* go together with a predicate. Such as for verbs we are pointed explicitly to which NPs and PPs could take part in the action denoted by the verb, i.e possible subjects, objects etc. This goes not only for verbs, but also for those nouns and non-finite verb forms that denote actions.

2.3.2 Previous work

As explained, the step from grammatically parsing natural language to a full semantic representation is as of today not practically computable. It seems either the demand of robustness cannot be met with or doing so would require exponential time [7]. So what do we do? One option is to give up, at least until the perfect parser turns up. A perhaps more attractive option is

[6]Christof Monz, christof@science.uva.nl, is a PhD student for my supervisor Maarten de Rijke

[7]I mean that the number of interpretations is exponential in the number of ambiguities ⇒ exponential time, unless we can get rid of the ambiguities

Categ.	Chunk	Semantic Head	Dependencies	Other
NP	BELFAST	BELFAST		**Province**
,	,			
NP	Northern Ireland	Ireland		**Province**
(	(			
NP	AP	AP		
)	)			
NP	The leader	leader	AP, Northern Ireland, Northern Ireland, biggest Protestant party	
PP	of Northern Ireland	Ireland		**Province**
POS	's			
NP	biggest Protestant party	party		
VG	narrowly won	won	biggest Protestant party, Northern Ireland, a crucial party battle, Saturday	**Active Voice**
NP	a crucial party battle	battle		
NP	Saturday			**Time**
,	,			
VG	keeping	keeping	Saturday, a crucial party battle, the province, power-sharing government	**Active Voice**
JJ	alive			
NP	the province	province		
POS	's			
NP	power-sharing government	government	the province, Saturday, his coalition partners	
CC	but			
RB	only			
IN	by			
VG	promising	promising	power-sharing government, the province	**Active Voice**
TO	to			
VG	punish	punish	his coalition partners	**Active Voice**
NP	his coalition partners	partners		
SENT	.			

Figure 2.3: Text parsed into chunks. Note: (I) the adverb 'narrowly' nicely is incorporated in a VG (II) the parser leaves treatment of the "unbound" adjective 'alive' to the next level of analysis (III) 'but only by promising' was also a bit difficult and is handled conservatively

to scrutinize the actual goal task to see whether it really requires this sacred robustness.

The same situation has already presented itself in Information Retrieval (IR). To handle more specific requests, such as wh-questions in a Q/A-system, requires inference beyond that of simple pattern matching on words.

So when these IR people had to face the unattainability of a full and robust semantic representation, (they didn't give up) rather they conceived some intermediate rough-style NL text representation. The key being that this representation should capture the *who-did-what-to-whom-and-when* content of each sentence - which would be enough for the purpose of answering such who-what-whom-when questions. Various *Description Logic*s i.e languages for representing knowledge have been proposed but typically formulas would look like (1) and (2) [Monz, 2000].

1. a. Industry Sources put the value of acquisition at $ 100 million

   b. $\exists agent.(source \sqcap industry) \sqcap \exists event.put \sqcap \exists patient.(acquisition \sqcap \exists value.(100,000,000 \sqcap dollar))$

2. a. John Blair was acquired last year by Reliance Capital Group Inc.

   b. $\exists agent.reliance\_capital\_group\_inc \sqcap \exists event.acquire \sqcap \exists patient.john\_blair \sqcap \exists time.last\_year$

For more information on Desciption Logics in general we can refer to [Donini et al., 1996], and for DLs actually used in Information Retrieval [Litkowski, 1999] and [Meghini et al., 1993].

Apart from representation *extraction* considerations, having first-order logic as your representation means both good and bad things. It's good that reasoning with first-order logic is well studied and tools for that don't have to be invented. On the other hand one can expect potential cases involving a too high computational cost and answers will necessarily be in strict yes/no form. [Blackburn et al., 1999] however, present an example where first-order theorem provers and model generators can be of practical use for certain linguistic problems.

On the reasoning side, one can and should fall back on the vast research experience there is on efficient automatic theorem provers. However, in the NL setting scalability is a persistent problem. Indeed, [Monz, 2000] concludes the use of various popular theorem provers[8] on standard data collec-

[8]The ones in question are Bliksem [BLIKSEM, 2000] and SPASS [SPASS, 1996]

tions like TREC [Harman, 1995], consisting of thousands of documents, as 'computationally very challenging' (cf. [Crestani et al., 1995]).

2.3.3 Our alternative

As an alternative, we propose reasoning with much lighter representations, the reasons for which are worth repeating

- light representations are quick and easy to generate
- presumably reasoning with them is slim as well
- the inherent uncertainty has the advantageous side-effect of yielding *degree* answers rather than boolean.

The downside is of course the information that is lost in the simplification process. However, with a meaningful evaluation procedure we can investigate exactly what happens when we relax the pressure on semantic faithfulness. It might be sufficient depending on the particular need. Indeed, many applications ask only for approximate answers or prefer them to no answers at all. Also, one of the key aims with this thesis is find out how much, if any, of the semantic information can be recovered and used, without stepping into the computationally costly area.

To be more concrete, we shall try two different shallow approaches to entailment computing. One very light, pursued and explained in chapter **3 Minimal Representation Tests** on p.26, and one deeper, yet still fairly light in absolute terms, in chapter **4 Richer Representation Tests** on p.56. We are also interested in the relation between them.

In any case, I hope lines of research with the traditional heavy analysis are performed in parallel and can be compared to those pursuing the shallow representation (like in this thesis). Quite possibly, they will be superior to one another on various tasks with *different* demands.

2.4 Evaluation

As is often repeated, natural language processing and semantics is an area full of interesting ideas that may or may not turn out to work well in practice.

Therefore, it is of huge importance to have a sound evaluation procedure. Usually that consists of

- A large text collection
- Keys to cases one wants to test on. Being in NLP, typically a human has to sanction the test cases
- Relevant performance measurements

We will go through our particular set-up step-by-step.

2.4.1 The set of newspaper articles

The text collections consists if a small corpus of 69 news stories from **AP** news wire, **BBC**, **CNN**, **L.A. Times**, **Reuters**, **USA Today**, **Washington Post** and **Washington Times**. This collection was categorized (by hand) into 21 topics. Within one topic, all documents were released on the same day and describe the same event.

The documents were segmented into paragraphs, which in newspaper articles tend to be short. Since computerized paragraphing has been shown to be quite successful [Hearst, 1994], this could be applied to fuse some of the authors' original segmentations. The method is along the same lines as this project, i.e. taking advantage of the simpleness and domain-independentness of *idf word-statistics*[9] and then see how it matches to human judgments, and was already carried through by Christof Monz when I got my hands on the collection.

Figure 2.4.1 shows each segment of two news stories covering a plane crash in Taiwan. Now, looking for entailments, we see that **AP 2** is as least as informative as **Reuters 4** and as **Reuters 5**. But for instance, although **AP 2** is large and has a lot of information, it does not include the specific information mentioned in **Reuters 3**.

Each topic has on average 3.3 documents telling that news wire's version of the story and one document on average consists of 16.4 segments (More statistics on the average documents is shown in figure 2.4.1). That means we have to check roughly 55.9 over 2 possible entailment relations for the average topic.

[9]Explained in chapter **3 Minimal Representation Tests**

Reuters (31 Oct 2000 16:39 GMT)	AP (31 Oct 2000 16:25 GMT)
Reuters 1: TAIPEI (Reuters) - A Singapore Airlines plane bound for Los Angeles crashed during a typhoon at Taiwans international airport on Tuesday, an airport police official said. **Reuters 2:** It was not immediately known how many of the 159 passengers and 20 crew were killed or injured, Civil Aeronautics Administration deputy director Chang Kuo-cheng told reporters. "The plane burst into flames and exploded shortly after takeoff," an airport police official told reporters. **Reuters 3:** Local television was reporting that over 120 injured had been taken to hospital. **Reuters 4:** The SIA Boeing 747-400 was taking off during a storm and hit by strong winds. It hit two other planes on the tarmac, including a China Airlines plane, police said. A Taiwan vice transport minister said no one was on board the other two planes. **Reuters 5:** The injured were rushed to hospital. No other details were immediately available. ⋮	**AP 1:** TAIPEI, Taiwan (AP) - A Singapore Airlines jetliner bound for Los Angeles crashed on takeoff in a storm Tuesday night and slammed into another plane on the runway, a Taiwanese official said. **AP 2:** There were 179 people on board Singapore Airlines Flight SQ006, which local media reports said was a 747. It was not immediately known how many people were hurt or killed, but local media reports said some injured people were being taken to the hospital. Strong winds seemed to have forced the plane down. There was an explosion as it struck a China Airlines plane on the runway at Taipeis Chiang Kai-shek International Airport, emergency official Wu Bi-chang said. Local media reports said the China Airlines plane was empty. **AP 3:** The crash occurred at 11:18 p.m. local time, and rescue workers were being dispatched to the scene, Wu said. Minutes later, the flashing lights of rescue vehicles were visible on the wet tarmac. Local media reports said there was a fire on the runway after the crash but that it had been extinguished. ⋮

Figure 2.4: Two segmentized document excerpts (from Topic 6)

	average per topic
number of documents	3.3 docs.
document length	612 words
total length of documents	2115 words
length of longest document	783 words
length of shortest document	444 words
segments per document	16.4
total number of segments	55.9

Figure 2.5: Table 1: Statistics on the Test Collection (21 topics, 69 documents)

2.4.2 The human judgments on the set

Recall the human judge from section **2.1** (p. 7), in more detail, the judgments on each pair of segments were to be chosen from three different ratings, `no entailment relation`, `contains a substantial part of` and `has all the information of`, respectively. Lets look at some examples to illustrate these ratings; in Topic 6, the **AP** and **Reuters** documents listed in Figure 2.4.1 were selected for human assessment.

Score 2: In Topic 6, the implications **AP 2** $\rightarrow$ **Reuters 4** and **AP 2** $\rightarrow$ **Reuters 5** both scored 2, because all information in the segments on the right-hand side is present either implicitly or explicitly in the segment on the left-hand side.

Score 1: An implication **A** $\rightarrow$ **B** was rated 1 whenever **A** entailed a substantial subsegment of **B**. For instance, **AP 1** (in Topic 6) says nearly everything expressed by **Reuters 4** except for the type of the aircraft and the fact that it was empty.

Score 0: An example where no entailment is found is given by **Reuters 4** and **AP 2**: **AP 2** contains a significant amount of information that is not present in **Reuters 4**. In such cases a score 0 was to be assigned.

Surprisingly or not, entailment relations turned out to be quite scarce. Out of 12083 potential pairs, 501 (4.15%) were rated 1, and only 89 (0.73%) received a score of 2. Figure 2.4.2 shows the incidence of entailment relations between the document paragraphs.

Rating 2	89
Rating 1	501
Rating 0	11497
Total	12083

Figure 2.6: Count of entailment pairs according to the human judge

It may worry the reader that the test collections is not very large, and entailment relations are very uncommon (only found in less than 1% of the potential cases) - even though we would imagine we're in an environment favourable to produce entailment relations - concise newspaper articles about the same event. Another drawback is that the human judge on all these relations is one non-layman person. So, in actual fact, we could be measuring the mind of this one person[10] and not really entailment relations. My supervisor tells me however that he is about to work out a new much larger test set, judged by more than one layman. This would remove the objections entirely.

2.4.3 Use of the test set

Now, we give the computer a method and let it calculate an entailment score for all paragraph pairs within each topic. This computed score, at least in the methods discussed here, varies between 0 and 1. To make sense out of a whole bunch of scores wildly varying between 0 and 1, into the form of `entailment present` versus `no entailment present` [11] we set a *threshold* value. Hence, with a threshold value between 0 and 1 we have an interpretation of the computed values into `entails` (those above the threshold) and `does not entail` (those below the threshold). As shall be seen, there are more clever ways of choosing the threshold value than to simply leave it at 0.5.

Precisely how do we compare the computed entailment scores to the human gold standard? Intuitively we want our computation to

(1) Recognize as many real entailment relations as possible and at the same time

(2) Not falsely believe there are more entailment relations than the humans decided.

[10] Henry Chinaski, who deserves thanks for preparing over 12 000 judgments

[11] The more detailed investigator would perhaps map onto the three levels 'no entailment relation', 'contains a substantial part of' and 'has all the information of'

If it wasn't for (1) we could report no entailment relations at all and we'd be sure we haven't reported an entailment relation where there wasn't supposed to be one (although we'd miss the 500 or so real ones). If not for (2) we could respond positively in all cases and there would be no entailment relation missed (although more than 11000 mistakes).

The same situation arises in Information Retrieval where a retrieval algorithm is measured by how well it selects exactly those documents assessed by humans to pertain to a certain query. From there, the concepts *precision* and *recall* are defined as follows [Baeza-Yates and Ribeiro-Neto, 1999]:

$$Precision = \frac{\text{number of correct document retrieved}}{\text{total number of documents retrieved}}$$

$$Recall = \frac{\text{number of correct document retrieved}}{\text{total number of correct documents}}$$

Precision corresponds to (2) above and recall to (1). For our case, in order to be clear, we can reformulate

$$Precision = \frac{\text{number of correct entailment pairs computed}}{\text{total number of entailment pairs computed}}$$

$$Recall = \frac{\text{number of correct entailment pairs computed}}{\text{total number of correct entailment pairs}}$$

In our experiments, these ratios of precision and recall will depend entirely on

(I) how we set the threshold value and

(II) how we relate it to the 0,1,2-scale of the human assessments.

We could say that if an entscore is above the threshold it's correctly identified if its human score is 2. Or we could be kinder and say the human score could be either 1 or 2 (`contains a substantial part of` or `entails`), just not 0 (`no entailment relation`). As hinted above, the lower the threshold we expect recall to rise on the expense of precision. Because the more relations we consider correct we should at least find some more correct ones but also a lot of noise. Vice versa, if we set a very high threshold we only get a few correctly identified entailment relations that we are quite sure of - good

precision, but since we are so strict a lot of true entailments will be dismissed as not sure enough $\Rightarrow$ low recall.

So how should the threshold value optimally be set? If we simply set the threshold high or low precision gets higher or lower, but recall the opposite. But we have no reason to suppose one of recall or precision is more important than the other. In fact, we want both to be high and for this particular experiment there might be a threshold, that is not the middle, that gives optimal (in a simultaneous sense) precision and recall figures.

There are several ways to combine precision and recall into one measure [Baeza-Yates and Ribeiro-Neto, 1999], the most straightforward being the f-score, defined as the harmonic mean between the recall and precision values

$$\text{F-score} = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}}$$

This assumes a high value if and only if precision and recall are both high.

Now we do experiments and see for which parameters of (I) and (II) we get the best f-score. It is not cheating to test for the best f-score and then set the parameters according to that. We have no non-violable predefined ideas on where a threshold like this should lie. It's equally surprising or good/bad if the optimal turns out to be at 0.3 or if it's at 0.7 because this is an exploratory experiment in an unsure area like NLP. As long as the test set it large and statistically wide enough, we are ok.

However, as tests were carried out, using different thresholds (ranging from 0 to 1 in steps of 0.1) and with different configurations of the 0,1,2-scale, it became clear that the best f-scores were achieved if we require only that the human-score be > 0 for a computer score above the threshold. This matter is discussed slightly more in [Monz and de Rijke, 2001], but assume from now and on in this thesis that

$$\text{Human assessed } A \left\{ \begin{array}{l} \textbf{has all the information of } B \text{ (rat. 2)} \\ \textbf{contains a substantial part of } B \text{ (rat. 1)} \end{array} \right\} :\Longleftrightarrow A \text{ entails } B \tag{2.1}$$

or the equivalent

$$\text{Human assessed } A \textbf{ does not entail } B \text{ (rat. 0)} \Longleftrightarrow \text{A does not entail B} \tag{2.2}$$

and we want to explore if

$$\text{the } \textit{computed} \text{ value } entscore(A, B) \text{ is above the threshold} \Longleftrightarrow A \text{ entails } B \tag{2.3}$$

or at least how they relate.

2.5 Chapter Summary

- If A and B are two natural language text paragraphs, 'A entails B' iff 'everything that has been said in B has been said in A'.
- Entailment relations are used for Q/A, topic detection and tracking, dialogue systems and more
- Entailment relations are computed in the following manner:
  generate some logic representation → reason with logic → result.
- Deep analyses have problems with scalability both on the level of generating representations and reasoning
- Alternative: extremely light representations

  Advantage: Quick, easy & degree answers

  Disadvantage: Misses subtleties (and perhaps a lot more than just subtleties?)
- Evaluate by comparing with a large set of human assessments
- View computer results in terms of precision, recall and f-score

Chapter 3

Minimal Representation Reasoning

When I got my hands on the project, a baseline version of a minimal representation reasoning engine was already implemented by my supervisor Maarten de Rijke and his PhD student Christof Monz. Their work published as [Monz and de Rijke, 2001], in which they state that their implementation is rather a skeleton for computing semantic entailment relations and that there is ample room for

- Making it linguistically more informed
- Comparing it with similar reasoning schemes on richer representations

This is the justification for the chapter division and for the denotation 'baseline'; The **3.1 Baseline** section below describes the previous work by Monz and de Rijke as well the setting for various add-ons implemented by me in the **3.2 Beyond the baseline** section (p. 34).

3.1 Baseline

3.1.1 Idf word-statistics

The main work horse of the method is the use of *Idf word-statistics*.

1. The NL text is first run through a Tagger (as described in chapter 2.3.1 Tagging), and out comes the text as a sequential list of *lemmas* along

with part-of-speech information i.e tags which denote mainly word class (see chapter 2.3.1 Tagging).

2. Secondly, from being a sequential list of lemmas, we throw away all the grammatical structure between the words as well as the order between the sentences, and put all the lemmas, or *terms*, in one big bag.

In step 2, what we are doing will of course possibly and likely obscure **a lot** of semantic content way beyond recoverability. The key issue is, exactly how much is lost compared to how much simplified computation it accomplished - is it worth it? In which situations, if any, is it worth it?

Next, we observe that some words occur much more often than others. This can actually be exploited to distinguish between less and more important words in a given context and is something that has been sighted many times in Information Retrieval and NLP. For example, a word like *and* occurs in almost any text and cannot serve to distinguish any text in any context. However, if we have two documents about football and one of them contains, say, the word *devil* it seems one of the texts must be trying to say something quite unique. Words like *goalkeeper*, *game*, *fantastic* are likely to appear in a lot of documents about football and can't differentiate very well about what is meant in each document. But goalkeeper in a document in a different context, say religion, provides a salient feature of the presumably few documents that contain it. Note however that, in a bag-of-words representation we do not have the ability to tell if say *devil*, like above in the football context, occurs in a sentence like *the goalkeeper was playing like a devil to the attacking team*, then it simply means he played fantastic which is far from a salient statement in a football article.

Now, recall that we are interested in computing the relation between each segment of each document *under a certain topic*. Calculate the inverse document frequency of each term t_i as

$$idf_i = \log(\frac{N}{n_i}) \tag{3.1}$$

where N is the total number of segments in each topic, n_i the number of segments term t_i appears (no difference is made if t_i appears once or fifty times in one and the same segment) and the log is there only to smoothen the differences between the scores. We see here that idf_i, **I**nverse **D**ocument **F**requency, is higher for uncommon i.e more content-bearing words - which is the whole idea behind idf word-statistics.

This idea was borrowed (i.e. stolen) from Information Retrieval where it has been used, for instance, to measure the similarity between different documents [Baeza-Yates and Ribeiro-Neto, 1999]. It has gained ground due to its simplicity and, more importantly, because it's domain independent. We do not have to list beforehand what words are more and less meaningful for each possible topic. We just hand it whatever we choose as a topic and get the exact discriminating power of each word for that topic. The larger the body of documents get, the more we approach statistical safety and hopefully real semantic content-bearing, something which will be discussed further below.

3.1.2 Computing the entailment score

Let *d*, *d'* be two documents. Given the term weights as defined in (1), we compute the *entailment score*, $entscore(s_{i,d}, s_{j,d'})$, of two segments $s_{i,d}$ in d and $s_{j,d'}$ in d' by comparing the sum of the weights of terms that appear both segments to the sum of the weights of all terms in $s_{j,d'}$:

$$entscore(s_{i,d}, s_{j,d'}) = \frac{\sum_{t_k \in (s_{i,d} \cap s_{j,d'})} idf_k}{\sum_{t_k \epsilon s_{j,d'}} idf_k} \tag{3.2}$$

Intuitively: How much of the content-words in $s_{j,d'}$ occur in $s_{i,d}$?

The entscore as defined in 3.2 is the main value of interest, it's what we will use as the value of how strong an entailment relation we find there to be between two segments. How to interpret the absolute value will we discussed later, but we can already relate some of its logical properties.

Obviously the entailment relation measured by entscore is reflexive but, in general, not transitive. I.e. It's not always that $entscore(A, B) \cdot entscore(B, C) \leq entscore(A, C)$. For example, consider two disjoint sets A, B then $entscore(A, A \cup B)$ and $entscore(A \cup B, B)$ are positive but $entscore(A, B)$ is zero. There is another reason why we shouldn't expect it to be transitive, namely that weights may be very significant. I.e. if $entscore(A, B)$ and $entscore(B, C)$ are, say, 0.5 then if $entscore(A, C)$ is much lower it may be because what B and C had in common was heavier than the intersection of A and B. An example from the test set, see figure 2.4.1, is $entscore(\mathbf{AP1}, \mathbf{Reuters2}) \approx 0.63$ and $entscore(\mathbf{Reuters2}, \mathbf{CNN5}) \approx 0.32$ but $entscore(\mathbf{AP1}, \mathbf{CNN5}) \approx 0.16$.

Antisymmetry is not that interesting to look at because, the entscores, derived through set intersections, carry all we can say in terms of identity

between two sets A, B. If $entscore(A, B)$ is say 0.7 and $entscore(B, A)$ is 0.8 we can say that 80% of the content of A is in B and 70% of the content of B in A, nothing more nothing less because it's plain set membership. However, whenever entscore is 1 we have both antisymmetry and transitivity.

Finally, we can see how idf word-statistics is a shallow representation that can be quickly generated. Entscore also provides the computationally cheap and non-boolean reasoning device sought after in the previous chapter. The complexity of the reasoning scheme is quadratic[1] (with high coefficients) in the number of documents in a topic, but remains tractable in practice. Even with three times[2] the average number of documents in a topic (which is 3.3), computing all the entailment scores of all segment pairs takes less than a minute on an 800 MHz AMD PC. The number of segments in a topic is linear (by a factor on average 16.4) in the number of documents and the number of paragraph pairs is quadratic in the no. of segments, and finally the number of summands in a paragraph has a constant bound in practice.

3.1.3 From text segment to numerical entailment score

Here is a sample of two segments (Reuters 1 and CNN 1 from topic 6) and their lives through the process described in 3.1.1 and 3.1.2:

Newspaper segment

Reuters 1:
TAIPEI (Reuters) - A Singapore Airlines plane bound for Los Angeles crashed during a typhoon at Taiwans international airport on Tuesday, an airport police official said.

CNN 1:
TAIPEI, Taiwan (CNN) – Singapore Airlines Flight 006 –a Boeing 747 – crashed during takeoff from Taipei's main airport late Tuesday with 179 passengers and crew on board, according to Taiwanese aviation officials. Initial reports say some people on board survived.

[1] [Monz and de Rijke, 2001] mistakenly write exponential

[2] Chosen so that it outnumbers the most extreme case that actually occurs in the test collection

Tagged

See figure 3.1 below.

Word	Tag	Lemma
TAIPEI	NNP	TAIPEI
(	(	(
Reuters	NNP	Reuters
)	)	)
-	:	-
A	DT	a
Singapore	NNP	Singapore
Airlines	NNPS	Airlines
plane	NN	plane
bound	VBN	bind
for	IN	for
Los	NNP	Los
Angeles	NNP	Angeles
crashed	VBD	crash
during	IN	during
a	DT	a
typhoon	NN	typhoon
at	IN	at
Taiwan	NNP	Taiwan
's	POS	's
international	JJ	international
airport	NN	airport
on	IN	on
Tuesday	NNP	Tuesday
an	DT	an
airport	NN	airport
police	NN	police
official	NN	official
said	VBD	say
.	SENT	.

Word	Tag	Lemma
TAIPEI	NNP	TAIPEI
Taiwan	NNP	Taiwan
(	(	(
CNN	NNP	CNN
)	)	)
–	:	–
Singapore	NNP	Singapore
Airlines	NNP	Airlines
Flight	NNP	Flight
006	CD	@card@
–a	DT	a
Boeing	NNP	Boeing
747	CD	@card@
–	:	–
crashed	VBD	crash
during	IN	during
takeoff	NN	takeoff
from	IN	from
Taipei	NNP	Taipei
's	POS	's
main	JJ	main
airport	NN	airport
late	JJ	late
Tuesday	NNP	Tuesday
with	IN	with
179	CD	@card@
passengers	NNS	passenger
and	CC	and
crew	NN	crew
on	IN	on
board	NN	board
,	,	,
according	VBG	accord
to	TO	to
Taiwanese	JJ	Taiwanese
aviation	NN	aviation
officials	NNS	official
.	SENT	.
Initial	JJ	initial
reports	NNS	report
say	VBP	say
some	DT	some
people	NNS	people
on	IN	on
board	NN	board
survived	VBD	survive
.	SENT	.

Figure 3.1: Reuters 1 and CNN1

Idf scores

We give them only for the words in the example segments, not all the words in the topic.

```
taipei 1.6094379124341
( 1.6094379124341
```

```
reuters 0.916290731874155
) 1.6094379124341
- 1.6094379124341
a 0.916290731874155
singapore 1.6094379124341
airlines 1.6094379124341
plane 1.6094379124341
bind 0.916290731874155
for 0.916290731874155
los 1.6094379124341
angeles 1.6094379124341
crash 0.510825623765991
during 1.6094379124341
typhoon 1.6094379124341
at 0.916290731874155
taiwan 1.6094379124341
's 1.6094379124341
international 1.6094379124341
airport 0.916290731874155
on 1.6094379124341
tuesday 1.6094379124341
, 0.916290731874155
an 1.6094379124341
official 1.6094379124341
say 0.916290731874155
. 0.22314355131421

cnn 0.916290731874155
@card@ 0.916290731874155
flight 0.916290731874155
boeing 1.6094379124341
takeoff 1.6094379124341
from 0.916290731874155
main 1.6094379124341
late 1.6094379124341
with 1.6094379124341
passenger 1.6094379124341
and 0.510825623765991
```

```
crew 1.6094379124341
board 1.6094379124341
accord 1.6094379124341
to 0.916290731874155
taiwanese 1.6094379124341
aviation 1.6094379124341
initial 1.6094379124341
report 0.510825623765991
some 0.510825623765991
survive 1.6094379124341
```

Entscore

The bag intersection is {taipei, -, a, singapore, airlines, crashed, during, a, taiwan, 's, airport, on, Tuesday, , , an, airport, official, said, .}.

$$\sum_{t_k \in A \cup B} idf_k \approx 23.71 \tag{3.3}$$

$$\sum_{t_k \in A} idf_k \approx 38.58 \tag{3.4}$$

$$\sum_{t_k \in B} idf_k \approx 50.75 \tag{3.5}$$

$$\textbf{Reuters1} \rightarrow \textbf{CNN1} \approx 0.47 \tag{3.6}$$

$$\textbf{CNN1} \rightarrow \textbf{Reuters1} \approx 0.61 \tag{3.7}$$

So CNN 1 was a little bit richer in content, whether they entail each other depends on the threshold. But at least if **Reuters 1** $\rightarrow$ **CNN 1** then **CNN 1** $\rightarrow$ **Reuters 1**. Both entail each other if threshold is at 0.3 or 0.4 as is optimal. According to the human judge both entail each other (score 2).

3.1.4 Baseline test outcomes

Figures of recall, precision and f-score as functions of threshold.

Figure 3.1.4 show the results in terms of recall, precision and f-score respectively.

- The highest f-score ≈ 0.346 at threshold 0.4.

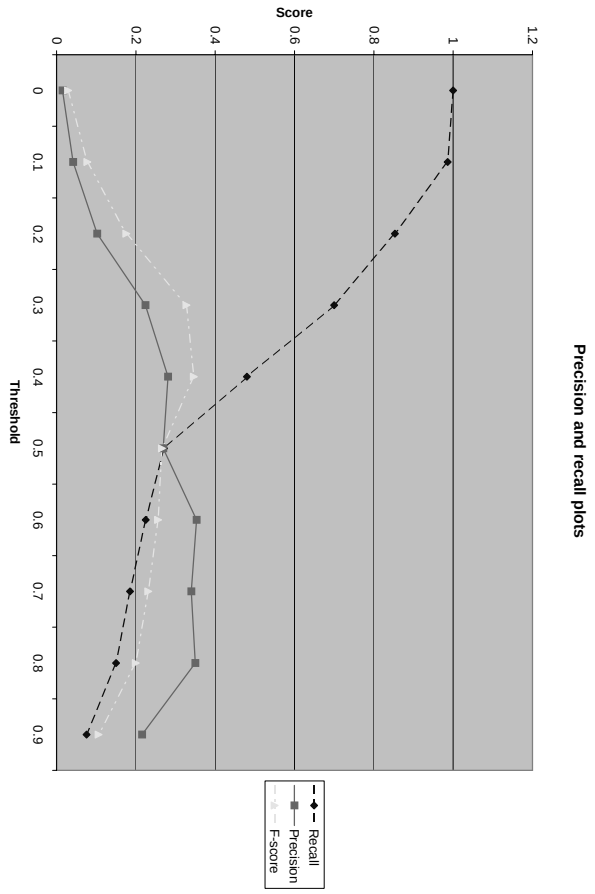


Figure 3.2: Baseline test outcomes

- That the threshold is better left at 0.3-0.4 (rather than 0.5) suggests that the method is comparably stronger on precision.

Further comments will follow in the **3.2.3 Conclusions on minimal representation tests** section on p.53.

As mentioned above, this framework (idf word-statistics and entscore as above) was already carried through [Monz and de Rijke, 2001] when I got my hands on the project. My task was to see if this approach had essentially reached its maximal power or if it could be extended further by incorporating some of the discarded linguistic information.

3.2 Beyond the baseline

There is a variety of possible enhancements, or to be more precise - it is conceivable that we could be cleverer, especially inspired by linguistics, in using the bag-of-words representation and that part is my original work on this project. Each idea will be presented, evaluated and discussed subsequently.

3.2.1 Bag-of-words level add-ons

Degrading stopwords

It's observable that some words, many of them occurring frequently, have little or no semantic content themselves but serve mostly to qualify other words. I.e, in linguistics, non-lexical words. For example, words like *is*, *the*, *for*. That a document contains the words *is*, *the* and *for* tells us virtually nothing about its semantic content. It could be about almost anything. Possibly we could get some edge on the entscores by treating these words differently.

The first task is to define more precisely which words, denoted stopwords, are devoid of own lexical content. On a domain independent level we can single out certain classes of words like particles, articles, temporal auxiliary verbs, grade adverbs to name a few. Note that we do **not** think that these classes of words are devoid of meaning when actually used in a text. There they certainly have grammatical meaning, and/or qualifying or discourse functionality. We mean only that in this idea where all grammatical structure between the words is discarded, they are *on their own* and thus without real ability to say anything.

In addition to whole classes of, when alone, less meaningful words there are a number of common words that have the same property but the rest of their class doesn't. For example, *case* is a noun but when alone it doesn't say much, *general* and *each* seem to be such adjectives too. There are various lists with words, not just thought to belong there, but statistically justified to. The stopwordlist employed in this project is given in Appendix A.3 on p.83. It's in courtesy of [Fox, 1992] and browsing it you will note that non-trivial words like *saw* and *important* are on it.

When implementing a stopwordlist filter we observe that there is a certain overlap between whole classes of stopwords like prepositions and articles - all the words of those classes do appear on the stopwordlist so maybe it's unnecessary to bother about a separate word-class based stopword filter at all. However, there may or may not be a degree of "stopwordness" related to word class in English. Also if we distinctly implement a stopwordlistfilter based on word class categories and one based on the list of words in **Appendix** A.3, that come from all word-classes, we can get a feeling for how liberal the stopwordlist is.

In service, the implementation was done as follows.

1. The words on the stopwordlist were all given a weight (for starters, one specific for all the words), in the interval [0..1]. The idf-score was then multiplied with that weight i.e. so that a weight of 1 corresponds to no change at all.

2. Each word in the document comes, after tagging, accompanied with its tag indicating basically word class. Each word class in then associated with another weight used exactly like the above, i.e. multiplication with a tag-specific degradation factor between 0 and 1. Different taggers use different tag sets[3]. There would have been more room for exploiting here with a richer tag set but no such Tagger was available and the results with the standard Penn Treebank[4] tagset are not encouraging enough to go seek a richer tag set.

   There are not so many different tags in the tagset so space allows me to list them here with a sample of weight assignments in figure 3.3.

   Finally, many experiments were carried out with different weights on

[3]See the section on tagging on p.11

[4]See **Appendix A.6** on p. 90

Tag	Weight	Tag desc.	Tag	Weight	Tag desc.
CC	0.001	Coordinating Conjunction	CD	1	Cardinal Number
DT	0.001	Determiner	EX	0.1	Existential there
FW	1	Foreign Word	IN	0.001	Preposition or Sub. Conjunction
JJ	1	Adjective	JJR	1	Adjective, Comparative
JJS	1	Adjective, Superlative	SENT	0.001	Full stop
NN	1	Noun, Singular or Mass	NNS	1	Noun, Plural
MD	0.05	Modal Verb	NNP	1	Proper Noun, Singular
NNPS	1	Proper Noun, Plural	PDT	0.0001	Predeterminer
POS	0.0001	Possessive Ending	PRP	0.01	Personal Pronoun
PRP$	0.01	Possessive Pronoun	RB	1	Adverb
RBR	1	Adverb, Comparative	RBS	1	Adverb, Superlative
RP	0.01	Particle	SYM	0.99	Symbol
TO	0.01	to	UH	0.8	Interjection
VB	1	Verb, base form	VBD	1	Verb, past tense
VBG	1	Verb, gerund or present participle	VBN	1	Verb, past participle
VBP	1	Verb, non-3rd person singular present	VBZ	1	Verb, 3rd person singular present
WDT	0.1	Wh-deteminer	WP	0.1	Wh-pronoun
WP$	0.01	Possessive wh-pronoun	$	1	Dollar Sign
WRB	0.1	Wh-adverb	:,()‘‘’’	0.001	Diacritics etc

Figure 3.3: Tags and sample weights (these are the optimal weights observed)

both the stopwordlist and tags. It would take too long to test all possible combinations in fine detail so I made a few restrictions, namely:

- The step size was usually no finer than 0.1
- Reasonable simplifications were done to sort out 'less interesting combinations', for example I didn't bother testing many configurations with the noun weight set low. More importantly, with the stopwordlistweight fixed, there are many combinations for the tag weights and pretty soon optimal looking weights schemes start to evolve. If say a certain scheme gave a lower f-score run on a stopwordlistweight of say 0.4 than 0.6, I wouldn't bother with testing on 0.2 although I lack a mathematical proof to guarantee the f-score would not be higher. Very aware of this, and since rigour is of utmost importance, I have tried to do a few more tests than what seemed reasonable.

The results in fig 2 show the following key points:

- The stopwordlist outperforms the tag weight filter
- The tag weight filter hardly has any significance at all when the stopwordlist is present
- Neither, nor together, provides a significant improvement. In fact, most weight settings gave slightly negative results.

WordNet synonyms

This is perhaps the most obvious way to improve on this system. One can say the same thing using different words, so in simplification, if we identify all words meaning the same thing and put them on a common index we will get a semantically fit representation.

WordNet [WordNet, 2002] is a great and free tool for getting quick access to synonyms in English. It is being extended to other languages and has not only synonymy relations but also hyponymy/hypernymy and more. There are several interfaces but one works just like desired, like a regular synonym dictionary, where a set of synonyms (synset) has a unique number which is used to index on concept.

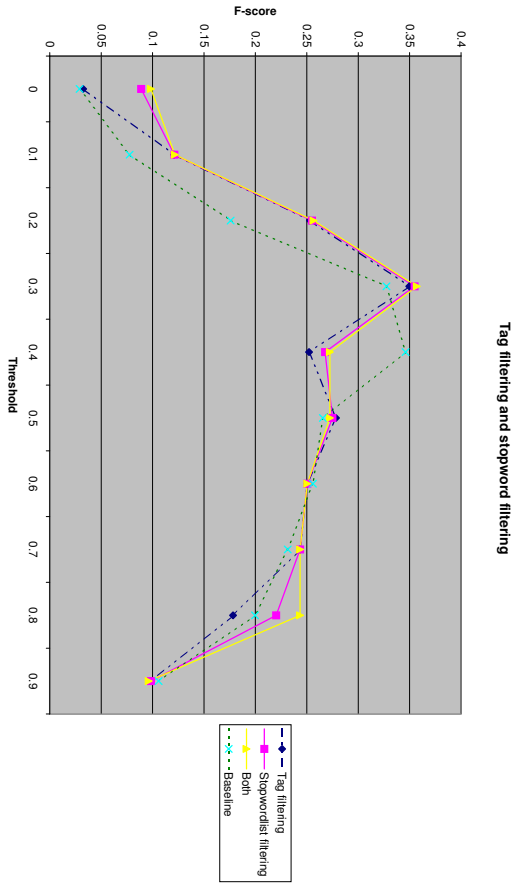


Figure 3.4: F-score as a function of threshold value. Top values are 0.349665 (tag-filter), 0.354853 (stopwordfilter), 0.356986 (tag + stopword-filter), 0.346008 (baseline)

The vast complication with the whole matter is that many words have more than one meaning i.e. several word senses. Such as *bank* in English means either 'financial institution' or 'riverbank'. The context suffices well enough for humans to distinguish the intended sense of a word but for computers to do this is a whole research area. It's quite common that words have more than one sense and the words with many senses are often common. In fact, the number of senses of a word is proportional to how common it is[5]! The more common a word is the more senses it has (although the relationship is not linear). For example, the verb *run* has 42 senses[6] according to [WordNet, 2002] Words in the WordNet database have on average 1.5 senses.

The science of determining the sense of a word in a given context is called *Word Sense Disambiguation* [Manning and Schütze, 2000]. A word like *bass* has two senses, that which has to do with music and that which is the name of a fish, that are clear and distinct. It's much harder to define the senses for a word like *way* and characterize its occurrences. Moreover, the fact that certain senses are more common than others makes things more complicated, esp. for evaluation. That the accuracy rates of various top-notch algorithms are lower than on other NLP problems shows that word sense disambiguation is relatively difficult. Suggested approaches include comparing the context of a given word to dictionary or thesauri glosses for each candidate sense, but like in many other cases in NLP, statistical methods seem to be the most popular. For instance, one can look at the surrounding few words of a word[7], and build a Bayesian argument for the most likely sense. However, the promised accuracy compared to work cost of implementing a disambiguator led me to choose not to do it for this project.

The tests run were quite disappointing (fig 2). Without any real disambiguation, we can for instance always select the most frequent sense since WordNet has this information. Unintuitively, slightly better scores were achieved with a more liberal implementation i.e that each word is taken to be synonymous with any word matching any of its senses, and, within the topic, any word one of whose sense matches

[5]This is known as Zipf's Law, but the refined relationship by Mandelbrot is more accurate [Manning and Schütze, 2000]

[6]These 42 senses are listed in Appendix A.4 on p. **??**

[7]Slogan: Know a word by the company it keeps

any sense of any word that has a matching sense. In other words, indexing on the union of the senses. Mathematically two words are on the same index ($\equiv$)

$$\forall a, b \text{ in a topic } a \equiv b \text{ iff } a = b \text{ or } \exists x \in syn(a), y \in syn(b)\ x \equiv y \quad (3.8)$$

Even if we bother to implement a more or less advanced disambiguator to rectify the behaviour of the indexing on concept, these results indicate there's not much to hope for in terms of f-score. But why are the effects apparently so weak when it seems like a quite good idea on paper? I believe the explanation lies in the fact that we are applying it on a too low level. The level of idf word-statistics is one where the structure between the words is absented. To a too large extent, the occurrence of a word somewhere in both of two segments is no guarantee they are of semantic entailment interest. It might say 'the president went by car to the meeting' is one but 'electric automobiles are expensive' in the other. Sure, with semantic indexing here, in a whole segment of text we get several extra matches like car = automobile but it appears not very many of them are useful. For it to be useful we'd like more of the words in the sentences to match, not just car but say president or electric or both. This is of course a standard objection to the whole shallow representation approach, but it's particularly relevant here because this is something that is added ontop and segments are not very long. If we have x matches without semantic indexing, and get say x+3 after it, it's not crucial, the x amount is still what's decisive. It's like installing a handicap-friendly elevator in addition to an already existing regular elevator in a three-floor building. Now the handicapped can ascend with ease but they are not in billions and the overall transportability of the building is largely unaffected.

As for hyponymy matches, i.e. that there is a match if entailed segment has a word which is a hyponym of the entailer, for example if something is said about Dublin, one can argue something is said about Ireland. Allowing for it would suffer from all the same deficiencies as synonymy as above, but to a much larger extent. Each word has a multitude of possible hypernyms, and the matches would be much less likely to be of semantic entailment interest. On a higher level, like in the second part of this project, we expect synonymy/hyponymy-considerations will play a greater role.

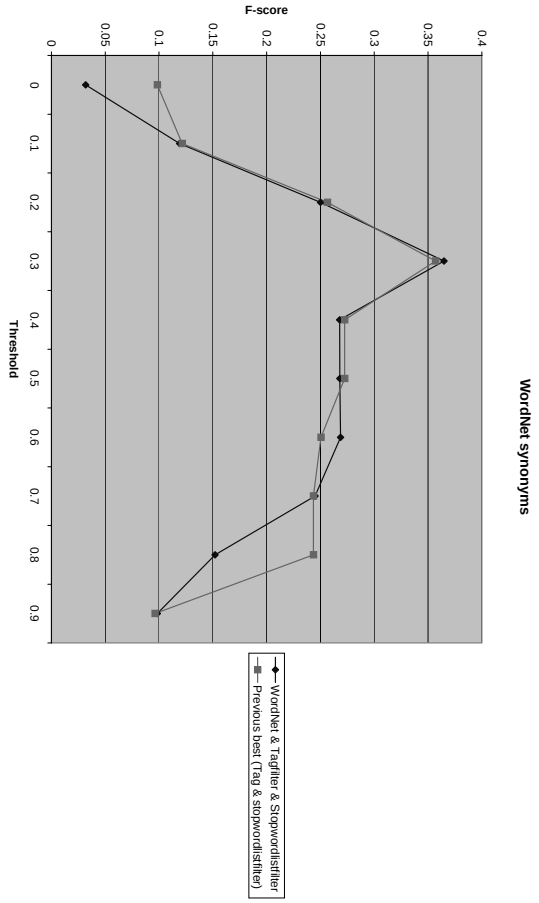


Figure 3.5: F-score as a function of threshold value. Top values are 0.356986 (previous best), 0.364853 (WordNet synonyms + tag & stopword filters)

Another possible source to the failure of recall to rise could be that, as noted in [Monz and de Rijke, 2001], entailment set by the judge at level 2, almost only occurs when there is an exact copy of the text[8]. So synonymy cannot possibly touch the recall on these.

Further ideas

Quite analogous to word sense disambiguation is *Collocations*. Word sense disambiguation is used to to split the same word meaning different things to increase precision. Identifying collocations is to fuse several words (usually bigrams though) standardly used with one meaning. I.e when there is some conventional way of saying "one thing" using several words, such as NPs like *strong tea*, *weapons of mass destruction* or phrasal verbs like *make up*. They conventionally bind together strongly in the sense that we cannot say for instance *?powerful tea*. For more information on collocations and computing them were refer to chapter 5 of [Manning and Schütze, 2000], but a fairly robust method is to simply go on frequency provided you train your collocator on a large test corpus. *Strong tea* and *New York* will be much more common than say *powerful tea* and *Old York*.

Although at least collocation identification can reach quite high levels of accuracy, word sense disambiguation is a more difficult task, these tools seem not to be freely available. Writing such tools from scratch is beyond the scope of this part of the project (at least if it requires proper evaluation of the accuracy) and since I cannot expect to do better than the state-of-the-art counterparts, there is no good reason to pursue it. It's likely to run the same fate as concept-indexing.

We do have some information for free from the tagger, namely that of word class. Some disambiguation can be achieved seeing for example a difference between saying (noun) and saying (gerund verb). When this is applied, wherever possible, it yields a puzzling slight decrease in f-score. Considering that disambiguating into word classes is not the same as disambiguating into meaning, this is not so puzzling anymore. Some words of different word classes such as hi-jack (noun) and hi-jack

[8] I myself can neither confirm or deny this since I wish to abstain from any inside information on when and why the judge set this or that level

(verb) should ideally be made to belong to the same concept whereas has (temporal auxiliary verb) and has (main verb) should not. Some experiments with re-fusing some chosen classes yielded no observable improvement, probably because this opposition holds a very obscure position in the full picture.

3.2.2 Pronominal anaphora resolution

Background on anaphora resolution

Anaphora is the linguistic term for general reference upwards, i.e to something previously mentioned, among NL text items (the terms for reference downwards i.e to something that is yet to be mentioned, which is less common, is kataphora). For instance

> George W. Bush is the president of the United States. He is sometimes likened to a monkey. Some people say "Look! Now the monkey did this or that" when he's done another something unwise.

As we understand English, the *he* refers back to the previously mentioned *George W. Bush*, in fact it's a shorter exact substitute for it. Why natural languages favour this kind of general substitute words instead of repeating is a question way beyond the scope of this thesis[9] - it suffices here to observe that English does it. *He* happens to be a personal pronoun and as such it refers almost always back to some previously mentioned and/or understood noun phrase, but also noun phrases in general are anaphoric depending very much on the discourse use. As we are apparently told in the above quote, the last use of *monkey* refers to George W. Bush.

It is not difficult to see that anaphora resolution in general is a giant step towards a full logic representation of NL. Applying robust anaphora resolution to any semantic reasoning problem is sure to aid

[9]For a general introduction, speculation and related questions, the interested reader can check chapter 11 of Talmy Givón's 'Syntax: A Functional-Typological Introduction Volume I', John Benjamin's Publishing Co., Amsterdam/Philadelphia, 1984

positively but beforehand we have no idea of how much. It could be insignificant or turn out to be worth its weight in gold.

Pronominal anaphora resolution is to identify the referents of anaphoric pronouns in a NL text. There is a widely known fairly robust algorithm for this in English, namely that of [Lappin and Leass, 1994][10] as well as a version of it by [Kennedy and Boguraev, 1996] which requires less linguistic parsing. The methods will be outlined below, and as shall be seen, my implementation will draw from ideas of both. The available anaphora resolution algorithms are surveyed concisely but comprehensively by [Zhang, 2002], and my choice of implementation source was simple, due to the limited parsing resources I had at hand. Also there seemed to be no performance gains by choosing otherwise. There appears to be no already implemented full-fledged anaphora resolvers for public use, although I wish also to thank Richard Evans for letting his MARS engine[11] have a go on my texts.

The two do not differ in their overhead structure, although that of [Lappin and Leass, 1994] treats detailed cases individually to a much greater extent. Nevertheless, the procedure is, for each anaphoric pronoun

- Find all possible antecedent NPs by looking back reasonably far
- Eliminate of those, antecedents that *cannot* be the actual referends using a *syntactic filter*
- For each of the remaining candidates, calculate a *salience* metric, i.e a score of likelihood of being the true antecedent
- The noun phrase with the highest salience wins

Reasonably far in my case means 7 NPs back, an empirical number, since an antecedent further back could never succeed in beating all seven others.

[10]This algorithm is actually meant to be general for a few more languages like German and Spanish

[11]MARS is the name of Richard Evans' anaphora resolver. Unfortunately the current implementation does not distinguish gender, which is quite crucial on the test set texts (but not on the texts MARS was designed for - technical manuals), therefore the results were of limited use. The **M**itkov's **A**naphora **R**esolution **S**ystem, developed at the world's anaphora epicentre in Wolverhampton, can be tried at http://clg.wlv.ac.uk/MARS/ (Accessed March 2002), and its method is that of [Mitkov, 1998]

A syntactic filter in [Lappin and Leass, 1994] consists of six rules to identify impossible antecedents. Most importantly, as in one of the six rules, an antecedent is impossible if it has number or gender disagreement with the pronoun. *He* can for instance never refer back to a plural or female antecedent. It is also claimed that a Pronoun P cannot be coreferential with a NP N if P is in the *argument domain* of N[12] (because then it has to be a reflexive pronoun), and a phrase P is in the argument domain of a phrase N iff P and N are both arguments of the same head. The other four are likewise fairly advanced syntactical criteria which are better left out here, they are however given verbatim in Appendix A.5 on p. 89. [Kennedy and Boguraev, 1996] does not state explicitly their employed heuristics.

The salience score in [Kennedy and Boguraev, 1996] is allocated as in [Lappin and Leass, 1994] except for two additions (also included here). Each candidate antecedent C adds up a salience score calculated as:

- 100 if C is in the current sentence
- 50 if C in the current context[13]
- 80 if C is the subject of sentence
- 70 if C is in an existential construction
- 65 if C is the possessor in a possessive construction
- 50 if C is the direct object of a sentence
- 40 if C is the indirect object of a sentence
- 30 if C is the complement of a preposition
- 80 if C is the head of a noun phrase (as opposed to embedded in a PP or another NP)

Of course the absolute values are not important, only their relativity to each other. The choice of salience factors and weights are justified both on linguistic-theoretical functional hierarchies [Keenan and Comrie, 1977] and the experimental results of both [Lappin and Leass, 1994] and [Kennedy and Boguraev, 1996].

[12] Condition 2, see appendix the part of [Lappin and Leass, 1994] that is given in A.5
[13] We refer to [Kennedy and Boguraev, 1996] to tell exactly what this means

The [Lappin and Leass, 1994]-algorithm boasts an accuracy of 86% (evaluated on computer manual texts), that of [Kennedy and Boguraev, 1996] 75% (on randomly selected open domain texts), so it would be satisfying to run either algorithm. However, as might have been noted, the algorithms make essential use of linguistic information that a chunk parser does not provide. Modern pronoun anaphora resolution is indeed **not** a shallow reasoning tool. It would be beyond the scope of this project to implement robust full-parsing analysis from partial parse, and simple ad-hoc generating has no easy means for evaluation and is thus scientifically uninteresting. A simplified clone, without use of the deeper linguistic information, would still deviate too much from the original so that the accuracy scores couldn't be invoked. The way out is then to observe that anaphora resolvers have a more straightforward, although still quite tedious to build, testing model. The test set contained 1310 anaphoric pronouns, to annotate by hand each correct antecedent, took about 24 hours.

Pronominal anaphora resolution implemented

The full list of what is treated as an anaphoric pronoun is given in tables 3.6, 3.8 and 3.7. A few more should count as anaphoric, notably *one* and *oneself*, and *this* and *that* when they are used nominally (the most complete list I've come across is in [Denber, 1998]), but they are not treated here because of the non-full parsing.

Now for the algorithm, as a syntactical filter, basically the following was implemented

- fairly robust matching of gender, animate/inanimate[14] and number features
- checks for pronoun in adjunct or argument domain of candidate antecedent
- check for pronoun in NP domain of candidate antecedent

[14]Sometimes argued to be a personal/impersonal distinction. It does not matter much since use is inconsistent. I can call a dog 'he' as well as 'it', but it's not clear (to me at least) whether I am necessarily conceding personhood to the dog in the former case but never in the latter.

Pronoun	Properties
myself	sing., animate
yourself	sing., animate
himself	sing., animate, masc.
herself	sing., animate, fem.
itself	sing., inanimate
ourselves	pl. animate
yourselves	pl. animate
themselves	pl.

Figure 3.6: Reflexive anaphoric pronouns. These are easier to resolve than the rest because they always refer back to the subject of the last previous predicate.

Pronoun Subj. form	Obj. form	Possessive form	Properties
I	me	my	sing., animate
you	you	your	animate
he	him	his	sing., animate, masc.
she	her	her/hers	sing., animate, fem.
it	it	its	sing., inanimate
we	us	our/ours	pl. animate
they	them	their/theirs	pl.

Figure 3.7: Personal pronouns, the possessive form require to be replaced by their antecedents + a possessive ending

And as for saliency:

- robust current sentence salience boost
- robust existential salience
- robust possessive check
- robust oblique check

Now it looks as if the salience and syntactic filter parts are implemented some as ad-hoc, some robustly and some not at all. That is correct, but it's not in focus how it's done, for all we care about, there could be a monkey or random matcher sitting inside the anaphora tool. What we are interested in is rather

Pronoun	Properties
who	animate
which	inanimate
that	*matches anything*

Figure 3.8: Relative pronouns, the parser has trouble telling these from interrogatives which indeed have the same form. In the case of *that* it is confused with the demonstrative adjective use of *that*

- *How well* is the anaphora resolution done?
- How does it *help* (hinder?) computing semantic entailment relations using idf word-statistics?

Since we have a test set to compare with, we can answer the first question without scrutinizing the inside of the anaphora routine. The latter question is examined through the regular entscore-f-score procedure, where the entscore is computed over anaphora substituted segments.

Before giving the exact figures of anaphora performance, we must make an important note; Not all anaphoric pronouns always have a specific previously mentioned noun phrase to refer back to. Different types of such are explained and exemplified below where we explain shortcomings of the anaphora algorithm. The performance assessment is given in figure 3.9 we can see where things go wrong since the resolves records the reason why the correct antecedent wasn't chosen.

Reason	Number	
Correctly resolved	**610**	**47%**
Salience failed to point to the correct ref.	188	14%
Computer thought there was wrong agreement	80	6%
Reflexive Pronouns wrongly resolved	10	1%
Computer thought the was no antecedent (when there was)	355	27%
Correct antecedent rejected because in the arg. domain of P	67	5%
Total	1310	

Figure 3.9: Results of the anaphora resolver

The shortcomings of the anaphora algorithm can be summarized as:

- Wrong agreement: The parser, or perhaps more accurately the tagger does not identify gender or animacy of a noun phrase very well.
  - A lot of times proper names are tagged as place names and not as persons. Noun phrases[15] and personal names are not genderized[16].
  - Noun that refer to persons like *leader*, *official* or even *spokeswoman* are in practice tagged the same as inanimate *table*.
  - Also plurality matches fail occasionally on collective nouns - *delegation*, *party* and *cattle* failed to match they.
- Lack of a specific referent (this is common; 321 out of 1310 $\approx$ 25%):
  - So-called pleonastic use of *it* - It is often used as some general referent to something to be supplied from the context or something quite non-specific[17]. Here are a few examples that occurred in the test set: *work it out*, *made it to safety*, *it was rumoured*, *make it clear that*, *it's not known*, *it was X who*
  - Also quite frequently occurring is the use of *we* in quotes. A news story typically has a comment from various spokespersons of the article part-takers. For instance, The Prime Minister of Israel says "We are ...", where it is not mentioned explicitly who *we* refers to. So a computer, contrary to humans who understand the text, finds no matching antecedent.
  - Sometimes the pronouns point ahead (i.e kataphora), such as in *During his visit, Clinton ...* and *With Gore urging them on, officials ....* One could argue also that some of the uses of *it* labeled above as pleonastic are better understood as kataphoric.
  - Parser shortcomings[18] - Quite frequently, the parser does not identify NPs correctly. We have for instance <Kyoto> <Agreement>

[15] One could use WordNet [WordNet, 2002] for this

[16] One could use US census data for this since most of the names are English

[17] [Lappin and Leass, 1994] has a quite extensive list of heuristics to identify these uses of it

[18] This statement is not to express dissatisfaction with the parser, but to describe when anaphora resolution fails

(not <Kyoto Agreement>), <Now>, <that he> listed as NPs. Sometimes it even crosses sentence borders, i.e <despised. He> and <list night. He> are NPs. A feature of statistical parsers, as different from symbolic approaches, is that we do not get really uniform results. We have for instance sentence initial <Jeb> <Bush> (two NPs) and <Nguyen> <Thanh Luong> (admittedly a difficult case) but also <The Prime Minister's first lady Hilary Rodham Clinton> and <Circuit Court judge Jorge Labargal>. Both the human and the computer anaphora resolver had to choose only among parser-considered NPs, which to some extent must impoverish its impact.

  - The use of partial parsing as such. For instance, in 67 cases (5%) the argument domain rule filters out the correct antecedent [19]. There is nothing wrong with the rule but the parser is not aimed at providing the exactness it relies on. A very common example is "The president said he .." where *he* is thought of as in the argument domain of *said* whereas in reality it's the subject of a subordinate clause "that he ..". We don't try to guess where there's a left-out *that* so we have to live with this imperfection (disabling the rule all together causes a drop from 610 down to 296 correct resolutions).

- The salience metric failed to highlight the correct antecedent

From these figures we suspect that there actually is basis for arguing that the anaphora resolver is equivalent to a randomized monkey, for only 610 out of 1310 ($\approx 47\%$) were correctly identified. We might even suspect a lot of these 610 were just lucky shots. However, tests with random selection fall one step lower. Random selection from possible NP antecedents in the paragraph gets about 230 ($\approx 18\%$) right, and with the syntactic filter about 450 ($\approx 34\%$) right. Matters of improving the resolution algorithm would of course have been investigated further, if it wasn't for some striking results on the entailment computing side.

First applying the computer's suggestions for antecedents i.e by substituting away all the pronouns in favour of lexical NPs, and then calcu-

[19] It is of course not certain we'd make a correct resolution if it didn't be here we automatically get no chance

lating the entscore as usual, we end up with a very clear performance boost. From the previous peak at f-score about 0.36, we rocket up to 0.42! A 17% improvement. The full curve is shown in fig 3.2.2.

Immediately we start to wonder about the maximal impact of anaphora. What if we, just to see the limit, apply the human suggestions for antecedents instead. This is not meant for practice, since we are not meant to have human assessments at hand, but it shows the hypothetical limit, since by definition no anaphora engine can perform better than the human key.

The puzzling fact is that perfect[20] anaphora resolving only gives rise to *slightly* better entailment results. I.e it does not seem to be of great value that a *correct* NP antecedent is substituted, it seems to be of more value that there is a substitution, with just some NP in the paragraph, *performed*.

In regard to this, one is tempted to substitute all pronouns, that is, even the many with no clear antecedent with just some random discourse referent in the paragraph. But doing so would require to take a stand in the philosophical question on whether we want results that we cannot explain. To be more precise, it does not make sense to just substitute pronouns that are not truly referential with just something random, then we might as well choose to simply randomly double occurrences of various NPs until we get an improved f-score. Since, if we did this and it did improve the f-score somewhat, we could only justify it with the argument that 'it's good because (and only because) it happens to better the scores'. It would be better to work out a strategy with proper basis on something like 'looking for discourse important NPs'. Actually, going in the other direction seems easier, i.e work out a theory on 'discourse important NPs' by investigating which NPs, if tried as very discourse important, are the ones that yield better entailment scores.

I rather believe that the power lies in that there will be more variety in the idf-scores. For two reasons:

[20]Perfect only in the sense 'The parser has shown what it believes to be NPs. If there is of these an explicit NP antecedent, the human has selected the right one. If there isn't an explicit NP antecedent, no substitution has been performed'. Real perfect anaphora would be to substitute a semantically perfect NP for each anaphoric pronoun, regardless if there is such an NP explicitly mentioned in the text

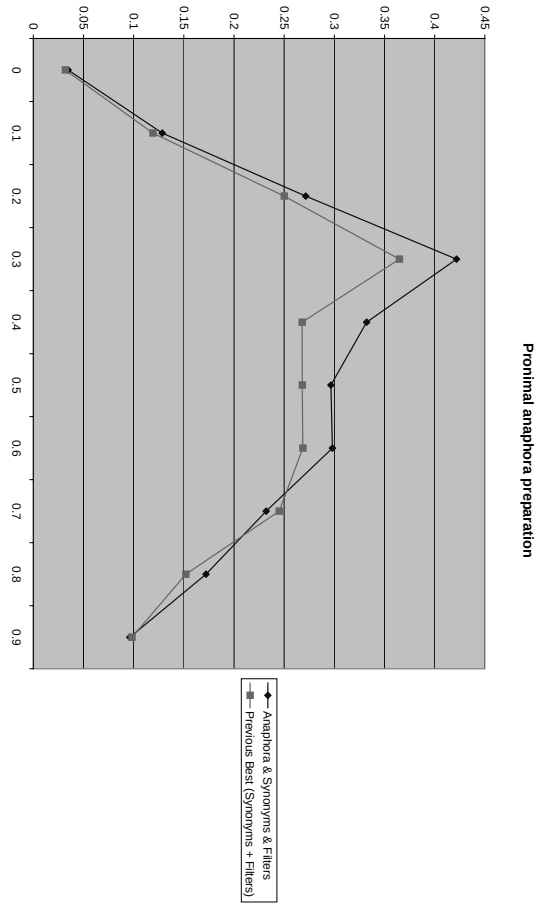


Figure 3.10: F-score as a function of threshold value. Top values are 0.421777 (Anaphora + everything, 0.364853 (Previous best)

- In a regular paragraph there is typically 1.1 ($\frac{1310}{21 \cdot 55.9}$) anaphoric pronoun. It's hard to believe substitution of that makes all the difference (esp. when in 25% of the cases there'll be no substitution.
- On each topic there's about 62.4 ($\frac{1310}{21}$) (minus 25%) substitutions. This will cause an alteration of the idf-scores. If we look at the idf-scores in the example in 3.1.3 on p. 30, most words have very similar idf-scores corresponding to 2,3-or so topic occurrences respectively, so there's lots of room for more discriminatory power.

3.2.3 Conclusions on minimal representation texts

(a) Idf word-statistics almost exhausts the power of the bag-of-words representation. We do not succeed in strengthening it by using word-level linguistic techniques.

(b) There is a significant improvement in preprocessing the texts with pronominal anaphora resolution and substitution. This anaphora resolution is properly *not* a tool of shallow reasoning let alone on bag-of-words representation.

(c) Idf word-statistics owes its strength to its natural approximating semantic-discourse linguistic phenomena. I.e it upgrades semantically salient semantic discourse items (on word-level) and downgrades less semantically crucial material, especially non-lexical words. More properly, these tests are the actual basis for saying that idf approximates that this or that word accounts for more or less of the content (although an even stronger alternative to devaluing words might be to substitute them and thus stepping towards more normal forms i.e it reduces the diversity that hinders comparison).

(d) We note that f-scores peak around threshold level 0.3-0.4. Even if applying extra linguistic tools does not improve the top f-score it would be an achievement if these tools could make the peak more flat. That is that say, we'd have consistently high f-scores from thresholds 0.3 to 0.6 instead of it going down straight after 0.4. That would make entailments to stand out to a larger extent in the total pool of non-entailments and entailments (or rather, we'd be

better at spotting them - which is the aim of these investigations). However, we can now empirically say that such results do not turn up and hence the extra add-ons do not provide such power.

(e) The top f-score of 0.42 corresponds to a recall of 0.60 and precision of 0.35. This means we can expect to find roughly 60% of true entailments, but only 35% of the positive answers we give are correct. Can we call this an *content* entailment computing device then, or have we computed some kind of *topic* entailment[21]?

(f) Also, what does in this mean in the full picture? If we want to take some random texts, run the engine on them and then look at the results expecting to find out pretty much which segments entail each other, we are dead wrong! Considering that entailments are scarce, simple probability[22] tells us that we'll get a **many** positive answers from the engine (because we are testing **many** cases), but most (65%) of these are just junk. We must still find the good 35% somehow, and what about the half (40%) of true entailments that went past the engine unnoted. This is the characterization of practical power of *entscore*, but I'll leave it up to the philosopher to say whether it's good or bad. As portrayed above it might seem more to the 'bad' side, but a good philosopher would also relate the difficulty of the problem into his/her rating. What kind of use it can serve however, depends on the task in hand.

3.3 Chapter Summary

- Idf word-statistics assigns different values of discriminating power to different words based on frequency within a topic.
- Entailment of A, B is computed by comparing the idf measured content of B that also shows up in A, to the full content of B.
- This gives a baseline top f-score ≈ 0.35

[21] [Monz and de Rijke, 2001] hypothetically paraphrase it as 'topic overlay', I prefer 'topic entailment' because overlay might be interpreted as symmetric. Entscore is not symmetric to any interesting degree.

[22] Assume say that there are 10 000 possible pairs, 5% (a reasonable figure) of which being true entailments. That's 500 true entailments whereof we expect 300 to be found (recall 0.6). Precision is 0.35 so we'd get about $\frac{300}{0.35} \approx 900$ positive answers

- One can imagine this can be improved on by adding some clever linguistic ideas that extract some of the meaning lost in the simplification to the idf-bag-of-words-model.
- Handling stopwords (words with little or no proper lexical meaning) and enabling synonyms helps the scores mildly
- This is disencouraging for further word-based exploitations like word-sense disambiguation and collocations
- We try pre-applying the texts with the non-word based NLP tool pronominal anaphora resolution. But we have only partial parsing to work on whereas anaphora resolution could take advantage of full parsing
- This turns out to be a 17% enhancement but it's more likely due to indirect discrimination effects on the idf-scores than the resolution per se

Chapter 4

Richer Representation Reasoning

It is difficult to draw on previous work along the same lines; Although there is considerable interest in Description Logics, putting it to use on this project requires the crucial step from NL to DL representation and here everyone uses different parsers and don't make their implementations publicly available.

Also phrase-based matching has been employed before, but likewise it is largely dependent on what kind of parses one starts working on. For instance, [Galbiati, 1991] present a mathematically well-argued article on phrase-based matching of database queries. However, the queries are not really natural language but in an unambiguous CFG that yields nice trees.

4.1 Chunks

As the next step to a more detailed representation of the NL input, we have chosen chunks because that can be done fairly robustly, so that we can blame any bad or good results of this entail engine itself and not on faulty parsing.

Recall from chapter **2.3.1 Parsing**, the chunk parser splits into mainly chunks of VGs, NPs and PPs; adverbs and verb-particles goes in the

VG. PPs are never decided to be included in a NP because of the non-triviality of knowing when a PP belongs to a certain NP or somewhere else (PP attachment problem)[1].

4.2 Text segment to bag-of-chunktrees

The parser suggests *dependencies* for all VGs. It does so also for some NPs of which the head happens to be an infinite form of a verb, and is thus potentially semantically equivalent to a verb. Dependencies are other NPs and PPs, which can be thought of as participants in the action denoted by the dependency holder. What kind of participant agent, patient, instrument, location etc we don't know but there are some heuristics. If there were a foolproof syntactical algorithm, a lot of NLP would be a piece of cake. Here's a sample sentence and partial parse (Topic 6, **UsaToday 3**).

> The crash occurred at 11:18 p.m. local time, and rescue workers were being dispatched to the scene, Wu said.

[1] A good paper on statistical approaches to this problem is [Hindle and Rooth]

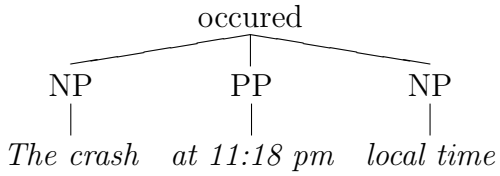
Category	Phrase	Dependencies
NP	<The Crash>	
VG	<occurred>	<The crash> * <at <11:18 pm>> * <local time>
PP	<at <11:18 pm>>	
NP	<local time>	
,	<,>	
CC	<and>	
NP	<rescue workers>	<local time> * <at <11:18 pm>> * <to the scene> * <Wu>
VG (passive)	<being dispatched>	<rescue workers> * <local time> * <at <11:18 pm>> * <to the scene> * <Wu>
PP	<to the scene>	
,	<,>	
NP	<Wu>	
VG	<said>	<Wu> * <to the scene>
SENT	<.>	

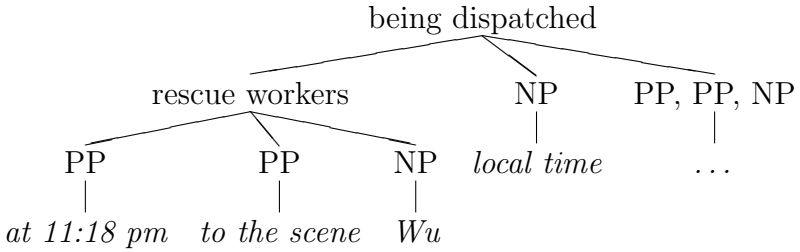
I.e parser informs, in addition to what happens to be correct, that it's possible 'local time', 'at 11:18 pm' and 'Wu' might syntactically belong to *being despatched* since it does not understand the meaning of the text. Moreover, the partial parsers does not exclude that 'to the scene' is a complement to *said* since it looks syntactically plausible. Most importantly, we have that the parser sees in the phrase 'rescue workers' the verb 'work', and hence it gives potentially that 'local time', 'Wu' etc might be participants in that action. That is, that without knowing the neither semantics of 'rescue workers' nor the context it's possible that the meaning should be something like 'workers of local time', 'Wu-workers', 'rescue work at 11:18 p.m' or 'rescue workers to the scene'.

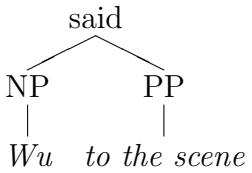
VGs (and some NPs) with dependencies, but who are not themselves dependencies of anything can be seen as roots. Since the partial parser merely gives *suggestions* of dependencies, these will not form beautiful tree structures. They look more like pseudo-trees since it can happen that a leaf has many parents, and some leaves can point back so as to form loops. If we eliminate loops (choosing whichever node as the

root), and split/clone all leaves having more than one parent we end up with real trees. One tree corresponding roughly to one NL predicate. This idealization of course comes with a cost of losing information as well as precision, but we'll see how much.

In the example above 'workers' has dependencies but it's itself a dependency to 'being dispatched' so we decide it's not a root. Roots are 'occurred', 'being dispatched' and 'said'. Thus we have three trees







Dependencies do not cross sentence borders (i.e the full stop symbol). Hence, in the chunk trees we already have a natural line of separation, and a text segment is now more like a sequence of chunktrees. If the chunktrees are "disjoint", the order of the sequence is not relevant. But in NL texts such an assumption of disjointness is certainly not valid - typically each sentences connects to previous and forthcoming sentences. Nevertheless, it's a necessary evil, which can be justified by the demands like

- The is no 'text segment normal form' or the likes - we have to chop up somewhere since paragraphs are too long for comparison,

and the sentence borders are, on the whole, the least bad place to do it.

- Order is in fact, at some level, irrelevant when it comes to narrating atomary facts. If I want to say A and B, it shouldn't matter if I say B then A or vice versa.
- Anaphora resolution performed before the chop up can lessen the damage

Finally, we have arrived at the desired representation, a segment is straight-jacketed into a bag-of-chunktrees. This is shallow reasoning, but arguably richer than the bag-of-words model. Things that are not NPs, PPs or VGs i.e interpunctuation signs, conjunctions, and any parsing "unresolved" matter is furthermore left out of consideration.

4.3 The Match Algorithm

The idea is to create a function that takes two representations of text segments A, B and returns a value between 0 and 1 that corresponds to the amount of information in B that shows up in A. We give the following recursive definition:

$$match(A, B) = \frac{\sum_{b \in B} max\{match(a, b) \mid a \in A\}}{\sum_{b \in B} 1} \text{ if } A, B \text{ are bags} \tag{4.1}$$

Intuitively, this tries to say, for each sentence (= a chunktree) of B, find the sentence of A that has the most of its information. This could theoretically be the same sentence for b:s, but that's just fine - if it has the sought information that's what counts. Compare the sum of the information of B:s sentences found in A, to the total sum of information in B.

A few remarks are in order, most importantly on the use of **max**.

- Why max? It seems to imply the existence of an entailment relation - we are looking for one and forcing one out even though we're not sure there is one. This is stronger than just a recall optimization or a linear forward of the threshold - it is an unsound

assumption of entailment existence. However, had the match-computation been an accurate calculation of entailment (as shall be seen this computation hardly gives an accurate idea of entailment or not), there would have been no problem in the use of max. Then, one could argue, that since a high value of match(a, b) indicates *justly* the presence of b:s content in a, max is the preferred selection function. Whatever comes out of the max would indicate a true presence of that much information, and since, by assumption, there is that much information it should be acknowledged. One could insert a threshold level filter here to get a theoretically coherent scheme. In this case however, the max is motivated by only one reason, namely that other alternatives (average, min, random) yield worse results.

- Long, short, information rich, information poor sentences (chunktrees) are all worth the same - a unit score[2].
- We need to check all combinations of $match(a, b)$ i.e all $a \in A$, $b \in B$, which takes high degree polynomial time (and quite time consuming in practice) and it gets worse. See the below for more comments on this.

On sentence chunktree level[3], match is defined analogously, if a, b are chunktrees:

$$match(a,b) = \begin{array}{l} W_{head} \cdot match(root(a), root(b)) + \\ W_{dependencies} \cdot \frac{\sum_{\beta \in children(b)} max\{match(\alpha,\beta) | \alpha \in children(a)\}}{\sum_{\beta \in children(b)} 1} \end{array} \tag{4.2}$$

Exactly the same remarks as above apply. The intuition behind it is that a chunktree a entails b iff the predicate of a (the head) entails that of b *and* for each argument of the predicate in b (the children) we have a matching (i.e entailing) one in a. a entails b to a lesser degree if one or more components doesn't have a match. The weight parametres W_{head} and $W_{dependencies}$ indicate how important the subparts should be. W_{head} and $W_{dependencies}$ should be interpreted as percentages that sum

[2]To improve on this one could for instance scale by $\sum_{t_k \in S} idf_k$, where idf_k is as in chapter 3.1 and S is a sentence

[3]NPs and VGs can have dependencies so with their dependencies they are chunktrees, PPs always have a NP embedded so they are too

to 1. We'll get to the point of the optimal assignments for these in the next section. One can argue that this algorithm is unnecessarily stupid or recall optimized since it tries *all* combinations of children, isn't it preferable to match the subject of a with the subject of b, the direct object (if any) of a with that of b and so on? Yes, it would be preferable, the problem being that parsers are not advanced enough to robustly identify the predicate roles of each dependent NPs. There is however an add-on modification to guess agent and patient with some simple heuristics which will be discussed further on. Another question is whether one should get any entailment points at all if the heads do not match; Does 'Adam is climbing' say anything at all about 'Adam is thinking'? They have Adam in common but the information they convey is quite disjoint, but for the purpose in this thesis it's an empirical question.

The remaining cases of $match(\alpha, \beta)$ where α, β is a root of a VG, NP or PP, are analogously as follows:

$$match(\alpha, \beta) = \begin{array}{l} W_{semantic\_head} \cdot match(semantic\_head(\alpha), semantic\_head(\beta)) + \\ W_{attributes} \cdot match(attributes(\alpha)), attributes(\beta))) \end{array} \tag{4.3}$$

Attributes on verbs are such things as tense, voice, aspect, modality, negation, adverbs and particles. The interrelative importance is regulated through variable weights.

The attributes of a noun phrase are such properties as definiteness, agreement features (animacy, gender, number), adjective attributes, genitive attributes (the parser does not have this information except in the case of possessive pronouns). The relative importance of these features, the importance relative to the head and whether you can any match at all if attributes match but not the head can be variably set.

PPs always consist of a preposition and a NP, thus their matching can be defined in terms of their NPs, α, β being PPs

$$match(\alpha, \beta) = \begin{array}{l} W_{pp\_head} \cdot match(preposition(\alpha), preposition(\beta)) + \\ W_{pp\_np} \cdot match(NP(\alpha)), NP(\beta))) \end{array}$$

You can match an NP with a PP, a VG with an NP and even a VG with PP since it spite of not matching syntactically they can semantically.

However, these cases are penalized by variable weights and it's of course difficult for a NP without dependencies to entail a VG with lots; it would score 0 in the dependency term etc

The matching of heads to heads is straightforward in the case of prepositions, simple string comparison. But with nouns, adjectives and verbs there's an elaborate synonym matcher. There are options to count any sense match as a match or to select a particular (the most common) sense for two words and then match them. For verbs, nouns and adjectives we also take hypernyms into account but with a variable penalty for each hypernym level step aswell as a maximum level depth to search. WordNet [WordNet, 2002] provides even more functionality, namely

For verbs:

- A predicate $ent(v_1, v_2)$ which is true iff v_1 entails v_2. I.e precisely what we need (however, this is predicate is defined on only 427 pairs)
- A predicate $cs(v_1, v_2)$ which is true iff v_2 is a cause of v_2. (defined on 225 pairs)

For an adjective and a noun:

- A predicate $at(n_1, a_2)$ true iff the adjective is a value of the noun for example (disposition, willing) or (colour, yellow) (defined 650 different pairs).

All of which are incorporated and subject to use under weight parametres.

The algorithm was chosen to be implemented in Python[4] for mainly software engineering reasons. The only drawback would be that Python is interpreting and in connection to that fairly slow executing.

In practice, although there is certainly room for a faster implementation, a run through all 12 083 segments takes about 50 minutes (!) on a 800 Mhz AMD PC, because of the combinatorial nature of the match algorithm.

[4]A high-level object-oriented functional programming language with popular bits and pieces from various other languages. Everything about Python can be found at www.python.org (accessed the 15th of March 2002)

4.4 Parameters to the match algorithm

The almost-full set of parametres are listed here, with comments and ideas of functionality.

4.4.1 Synonyms

```
GUESS_SENSE = NO # Always guess one word sense
UNKNOWN_WORD_SYN_VALUE = 0.01 # Word not in dictionary
IGNORE_WORD_CLASS = NO # Ignore word class in comparisons
ENT_WEIGHT = 0.5 # verbs
CS_WEIGHT = 0.5 # verbs
AT_WEIGHT = 0.5 # adj-noun
VERB_HYPERNYM = 0.7 # Weight factor for each level
VERB_HYPERNYM_MAX_DEPTH = 0 # Max number of levels to search
NOUN_HYPERNYM = 0.7
NOUN_HYPERNYM_MAX_DEPTH = 0
ADJ_HYPERNYM = 0.7
ADJ_HYPERNYM_MAX_DEPTH = 0
```

It's curious what hypernyms can do, still we expect a hypernym match to be weaker than a full match and that five or so levels up it's as weak as no match at all. The verbal relations *entail* and *cause* should be exploited as inscrupulous cases of entailment.

4.4.2 Verb phrases

```
#VG phrases \\
VG_HEAD_LEX_WEIGHT = 0.5 # A head match counts for half the total
VG_HEAD_TSM_WEIGHT = 0.1
VG_DEP_WEIGHT = (1 - VG_HEAD_TSM_WEIGHT - VG_HEAD_LEX_WEIGHT)

# What the TSM_WEIGHT is made up of
VG_TENSE = 0.1
VG_MODALITY = 0.5
VG_PARTICLE_ADVERB = 1 - VG_TENSE - VG_MODALITY
```

For sure, we are still dealing in very shallow representations so we expect tense, aspect and other details be very subordinate to dependencies and head lexeme - on the order of 10propagates the view that matching dependencies but not head should still count fairly well.

```
GUESS_AGENT_PATIENT = YES # Try to identify agent
                        # and patient and match those
#If GUESS_AGENT_PATIENT = YES then set these
VG_AG_PART = 0.5 # I.e matching the agent counts for 50%
VG_PAT_PART = 0.3
VG_ADV_PARTS = 1 - VG_PAT_PART - VG_AG_PART
# Yes means you don't get points for matching non-head
# stuff in a VG unless the heads match
VG_HEAD_IMPLICATIVITY = NO
```

To guess agent patient is the best hope for variation of performance of the match algorithm. The subject is taken to be the leftmost dependent NP and the patient the next leftmost NP or PP (On passive verbs the patient is the leftmost NP and the agent is a PP introduced by *by* if there is one). Conjecturing that this works quite well, on perfect parsing it should work almost always on NP predicates and most of the time on finite verb predicates (exceptions are can be for instance wh-questions and the newspaper frequent '"...." said mr X').

We don't give the technical details on how tense is calculated and matched, it suffices to say that they are mapped onto three levels 'past', 'present' and 'unbound' (infinitives etc).

4.4.3 Noun phrases

```
#NP phrases
NP_HEAD_LEX_WEIGHT = 0.5 # A head match counts for half the total
NP_ATTR_WEIGHT = 0.2
NP_NON_HEAD_ITEMS_WEIGHT = 0.1
NP_DEP_WEIGHT = (1 - NP_ATTR_WEIGHT -
                 NP_HEAD_LEX_WEIGHT - NP_NON_HEAD_ITEMS_WEIGHT)
# Yes means you don't get points for matching non-head stuff
```

```
# in an NP unless the heads match
NP_HEAD_IMPLICATIVITY = NO
```

On noun phrases, the situation is fairly similar. I would never think the head lexeme should account for less than 0.5, but this leaves little room for striking discoveries if dependencies, agreement features and lexical attributes are only to share 0.5. Among them however, again repeating we are thinking shallowly, dependencies should have precedence over subtleties such as determinacy and attributes.

We leave out the technicalities on the exact determinacy and attribute internal count, rest with a few remarks. Lexical attributes such as adjectives each one counts about twice that of determinacy or number. A perfect match with including explicit cardinal numbers however scores very high (= twice that of an adj. attr).

4.4.4 Prepositional phrases

```
#PP phrases
PP_LEX_WEIGHT = 0.5
PP_PHRASE_WEIGHT = 1 - PP_LEX_WEIGHT
```

We can draw here an vague analogy with people who experience difficulty when learning a foreign language to get the use of prepositions right. That it is often difficult, i.e there is difference, even for closely related languages suggests that preposition widths aren't semantically heavy. That foreigners who consistently use the wrong preposition aren't misunderstood suggests (not conclusively proves) that the exact preposition isn't very crucial. Thus I am ready to accept a preposition lexeme weight of less than 0.5.

4.4.5 Hyperchunkal weights

```
#Mixed phrases
NP_PP_PENALTY = 0.7
NP_VG_PENALTY = 0.3
VG_PP_PENALTY = 0.2
```

```
#Variation in dependency counts
DEPENDENCY_FAVOURISM_NP = 1
DEPENDENCY_FAVOURISM_PP = 0.7
```

The penalties for matching syntactically different phrases might seem like double penalty since they aren't likely to match very well anyway given their differences. Only tests will show which is the better. The so-called dependency variation is there to avoid the monotone +1 count for every dependent NP or PP phrase. We know from linguistic semantic hierarchy analysis that the most important semantic roles are patient and agent, not place and time. Patient and agent most of the time correspond to NPs and oblique roles are often coded by PPs. Therefore giving relatively higher importance to NP dependency matches is a reasonable approximation.

4.5 The match algorithm calibrated

Given the large number of parameters, it was not practically possible to test all kinds of combinations. The following restrictions were made:

- chunk *internal* parametres were assumed to be independent of each other
- we only test two or three values of each 0..1 variable, leaving finer tuning for later stages. After the results of tests with two or three fairly well-spread values we can scan the results for variables with interesting impact.

We can be more precise on what 'chunk internal parametres as independent' means. We model the world as there being an optimal configuration on semantic strength for the various constituents of NPs, and that this is independent of the corresponding configuration for PPs and VGs. Then we imagine there is an independent such distribution also on a hyperchunkal level, given that it works on chunks whose internal semantic value is optimally calculated. But we do not simplify so as to hold the *synonym* parametres independent of the NP, VG and PP variables.

Still the number of combinations of parametre values remains vast, and one run only, in the current implementation, takes almost one hour. The parametre calibrating needs only to be done once however, to find out pretty well how the optimal assignments should be and what kind of f-scores the calibrated algorithm yields. It is definitely not intended that the parametres be re-calibrated any time in future use.

Nevertheless, after a large series of runs (over 300), occupying a full computer room at school for several nights, all scores were gathered. The scores tentatively give information on two levels

- How the parametres look optimally for showing semantic importance
- How well we can perform entailment relation checking with the given method and text representation

The latter obviously appears much more relevant for the aim of this project but alas the results turn out to be devastatingly fruitless. The top f-score of all lies at 0.12, a fair sized chunk of configurations reach up to around 0.105 and a very large amount produce f-scores between 0.8 and 0.11. All have their top f-scores around threshold 0.7 which suggests that the method is quick to fill up recall. This is consistent with the heavy relying on the max-function to select matching phrases and confirms our intuition that the use of max forcibly assumes there to be entailment also when there really isn't. Also when pre-applying pronominal anaphora resolution like in the case with shallower representation, we get almost no variation of the scores.

We present a selection of some deviating configurations in fig 4.5.

Applying the appreciated anaphora resolution does not enhance the method. Pronouns, being (in) NPs, already match according to the same criteria as in the anaphora resolution algorithm and the extra precision one gets by selecting the *correct* antecedent is apparently negligeable in the context of the match procedure.

Finally, we can give some comments on the calibrated parametres, keeping yet in mind that such low f-scores detract from the utopia that these parametres show true discourse-semantic importance. The methodology for locating interesting comments among the ocean of numbers was the following:

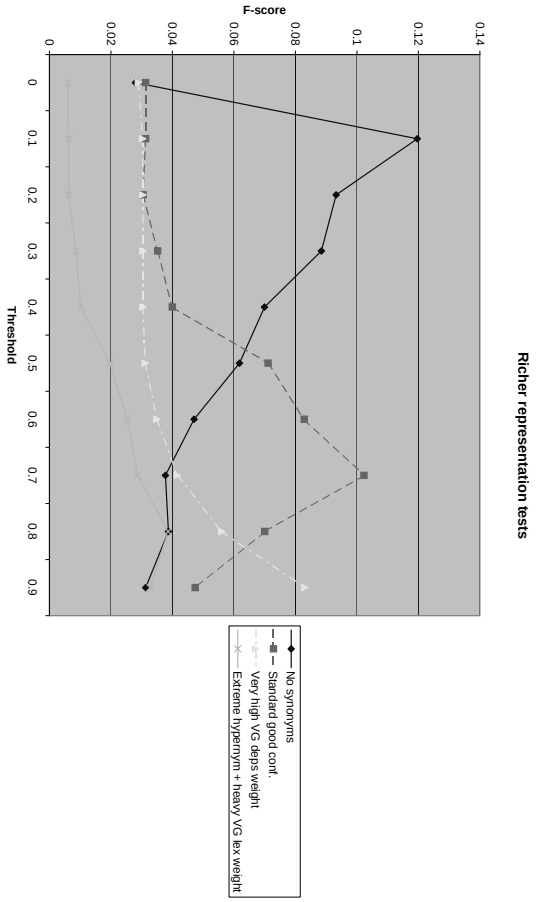


Figure 4.1: A sample of interesting configurations on the match-algorithm

(a) List the top f-scores for each configuration of an independent test series

(b) By eyeball comparison select the five or so tops and the five bottom and see what is in common

(c) Draw conclusion

How the weight are set in the NPs is what matters most the overall f-score, this is consistent with the high performance of the idf-approach relative to this argument-structure measure.

4.5.1 Noun phrases

- The head lexical weight shouldn't indeed be less than 0.5 but the attribute weight contra the dependency weight should be opposite my conjecture. Attribute should count for say 4 times as much as the dependency weight. The cause of this must be that the dependency scores are calculated through the naive 'select the *max* match'-heuristic which admittedly gives misleading scores.
- Points should be given for matching attributes and dependencies even if the head does not match (in the case of attribute weight and head lexeme weight high this does not matter much).

4.5.2 Verb phrases

The results here are a bit inconclusive because the top scores are achieved in cases where the, presumed important, head lexeme weight is either 0.5 or as low as 0.1. Nevertheless,

- particles and adverb should have low importance ≈ 0.1
- tense importance should be high ≈ 0.5, I can't justify this result unless it's because the dependency weight is forced low on account of it.
- it's clear that the dependency weight should be low for the same reason as with NPs.
- yet the idea of only giving points when there's a head match is wrong. Points should be given even if they are little.

4.5.3 Prepositional phrases

Not surprising is that 0.3 is better than weights of 0.5 and 0.7 for matching the preposition relative to the embedded NP.

4.5.4 Hyperchunkal weights

We are better off (but the differences are not big) penalizing quite heavily for mixed matching. I.e like this

```
NP_PP_PENALTY = 0.7
NP_VG_PENALTY = 0.3
VG_PP_PENALTY = 0.2
```

Mixed matching occurs in approximately 45% of the cases so the heavy penalizing can not be explained away by low frequency.

We get seem to get no help from favouring NPs among dependencies, again this has to do with the, in practice, subordinate position of dependency matches (perceived from many angles now).

4.6 Conclusions on richer representation texts

It is crystal clear that the f-scores are immensely low if we use the shallow idf-based method as a point of reference. We are frighteningly near the next lower point of reference i.e random which enjoys an f-score of around 0.03 for all thresholds (except at the very ends, 0 and 1).

We come to understand that the basic approach is simply so weak that seemingly relevant linguistic ideas aren't allowed to operate as intended.

4.7 Chapter summary

- With the help of a partial parser we convert the NL texts into chunks of speech i.e NPs, VGs, PPs, conjunctions, delimiters etc.

To able to compare segments we somehow need to transform to the form of a conjunction of propositions. Each chunk with a predicate is then simple-mindedly thought of as an independent proposition, hence we unite the chunks that go together (parser has this information) under one of them as head predicate. Commas, conjunctions, sentence order etc is subsequently ignored. A paragraph is now represented as a bag-of-propositions, propositions in turn being trees of chunks.

- With this representation we can compare segment contents by matching the propositions with each other. A segment A entails B if an enough amount of propositions in B have counterparts in A. To find this out without loss of recall we check all combinations of propositions in A and B and take the maximal (optimal?) matches.
- Matching a proposition with another is done by matching the main predicate and the predicate arguments recursively and according to various substrategies and weights.
- We do not know beforehand how to optimally set weights but we have a lot of linguistically motivated ideas
- We perform entailment tests that are evaluated according to the usual precision-recall-f-score measures. The combinatorial characteristic of the matching algorithm makes the whole procedure take a long time.
- The results unmistakably show poor performance (top f-score around 0.12) so we have to draw the conclusion that the whole matching approach is naive.
- Because of its low entailment computing power we are cautious to evaluate our ideas about weight parametres on it. Although huge amounts of parametre configurations have been checked, their impact is deemed to low to be conclusive.

Chapter 5

Conclusions and Further Work

Now we are ready to answer the first question from the introductory section, namely 'what is the trade-off between reasoning with lighter and richer representation?'. We conjectured that richer representation should come with at least as good performance on the cost of more analyzing work. That conjecture can be firmly decided in the negative and no rational being would want to trade more work and time for significantly poorer results.

We must find reasons for this evident state of affairs and the first one the blame is the very strong method of idf word-statistics. Secondly, it is apparently way too optimistic to think tree-matching is going to compare real semantic content (however symbolically beautiful it is). For example, even semi-advanced matching (or idf-word statistics) cannot touch that meaning which is beyond synonyms and syntactic likeness, that is cases like "I play football' implies 'I do sports' or 'X is the author of Y' is the same as 'Y was written by X' where the predicates are quite different.

It is unthinkable however that a tree pattern matching approach should not function, at least for precision, if it has less diverse data to work on. The problem is that real NL sentences are too seldom of the straight "A did X to B at time Y on place Z"-form; they include instead perhaps an anaphoric pronoun, then a relative clause, a predicate consisting

of a verb and a noun complement, in a newspaper article perhaps a quote with an ellipse, an adverb and then a temporal-causal subordinate clause. To tackle this robustly we must, as the most important steps, learn to

- partial parse towards perfection
- identify semantic roles of NPs and PPs
- split up into subordinate clauses, quotes and other stuff that detract attention from the main sentence predicate

Only afterwards Description Logics and normal forms become really interesting for entailment computing.

One important factor on why this isn't achieved yet appears to me to be that NLP suffers from deficient modularizing. In exaggerated terms, everyone is working with different parsers and tagsets, there must have been thousands of simple parsers independently implemented and that is surely a waste of resources. It shall be cured by people *making their NLP tools available* and that *suitable interfaces* are defined and put to use. Perhaps this is something NLP can learn from other research areas, if it's not sighted and acknowledged within how it tremendously prohibits efforts to join.

The idf word-statistics method is quite immune to linguistic add-ons on the bag-of-words level (I call this property *heavy*) and it seems idf word-statistics will stand at the top for quite some time since if we want to push it off we have to either

- Improve parsing
- Invent a clever heuristic corresponding to idf but on chunks or some other modestly richer level

However, we must not forget that idf word-statistics is dependent on there being a well-defined *topic*. In this case the topics were *human* divided, the non-idf approach does not have a corresponding addiction. Document categorization can be done with around 80% accuracy according to a survey paper by Yang [Yang, 1998], so this it not a big drawback. So enlarging topics to get more differentiated idf-scores should definitely be investigated.

Meanwhile, we have discovered that deeper linguistic analysis is sweet to use for pre-processing the texts, almost 20% sweeter[1], but the main reasoning tool shall remain the shallow idf-style. To get the maximum out of today we can invest in

- collocations
- word sense disambiguation
- pronominal anaphora resolution[2]

An interesting question is whether we can combine both methods, especially since one (idf) is comparably stronger on precision and the other has a high recall even on very high thresholds. We could for instance optimize on precision by saying descreeing (still varying thresholds):

A entails B iff A idf-entails B *and* A chunktreematch-entails B

i.e taking the intersection of the respective computed sets of entailments or, when being more interested in recall, the union

A entails B iff A idf-entails B *or* A chunktreematch-entails B

However, as noted, idf is almost always[3] better than the chunktreematcher and thus can learn very little from its clumsy big brother. More importantly, they're method are quite similar since it all comes down to matched words (this is as opposed to imaginary methods that could see entailment even on very different surface forms as in examples like 'Jane hates all four-legged animals' entails 'Jane does not love dogs'. So it's likely that for each threshold level the set

$$\{(a,b) \mid \text{a idf-entails b}\} \subseteq \{(a,b) \mid \text{ a chunktree-entails b}\} \qquad (5.1)$$

in which case the combination has no effect.

To sum everything up, if you want to improve entailment relation computing, improve other NLP tools instead!

[1] A 20% increase in f-score is equivalent with a 20% increase in both precision and recall

[2] For instance, [Orasan and Evans] claim they can decide the much useful animate/inanimate-NP question with 97% accuracy

[3] Except for recall on lower (0.0-0.3) thresholds, in which case the chunktreematcher tends to have ridiculous figures like 98% recall and 0.01% precision

Bibliography

[Lappin and Leass, 1994] , S. Lappin and H. J. Leass, "An Algorithm for Pronominal Anaphora Resolution", Computational Linguistics, 20:4, 535-561, 1994.

[Kennedy and Boguraev, 1996] , C. Kennedy and B. Boguraev, "Anaphora for Everyone: Pronominal Anaphora Resolution Without a Parser" In Proc. COLING 1996: The 16th International Conference on Computational Linguistics, Copenhagen, Denmark, Aug. 5-9, 1996.

[Blackburn et al., 1999] P. Blackburn, J. Bos, M. Kohlhase, and H. de Nivelle "Inference and Computational Semantics", in Proceedings IWCS-3, 1999

[Hearst, 1994] Marti A. Hearst, "Multi-Paragraph Segmentation of Expository Text", to appear in ACL '94, Las Cruces, NM.

[Barzilay et al., 1999] R. Barzilay, K. McKeown, and M. Elhadad, "Information Fusion in the Context of Multi-Document Summarization", in Proceedings of the 37th Annual Meeting of the Association of COmputational Linguistics (ACL '99), 1999.

[Gardent and Webber, 2000] C. Gardent and B. Webber, "Automated Reasoning and Discourse Interpretation", Claus Report 113, Computational Linguistics, University of the Saarland, 2000.

[Mitkov, 1998] R. Mitkov (1998), "Robust pronoun resolution with limited knowledge", Proceedings of the 18.th International Conference on Computational Linguistics (COLING'98)/ACL'98 Conference. Montreal, Canada.

[Orasan and Evans] C. Orasan and R. Evans (2001), "Learning to identify animate references". In Proceedings of the Workshop on Computational Natural Language Learning (CoNLL-2001). ACL-2001.

[Kamp and Reyle, 1993] H. Kamp and U. Reyle, "From Discourse to Logic", Dordrecht: Kluwer, 1993.

[Radev, 2000] D. Radev, "A common theory of information theory from multiple text sources, step one: Cross-document structure", In Proceedings 1st ACL SIGDIAL Workshop on Discourse and Dialogue, 2000.

[Harabagiu et al., 1999] Sanda M. Harabagiu, Marius A. Pasca and Steven J. Maiorano, "Experiments with Open-Domain Textual Question Answering", In Proceedings of the COLING-2000. Association for Computational Linguistics/Morgan Kaufmann, Aug 2000.

[Schmid, 1994] H. Schmid, "Probabilistic part-of-speech tagging using decision trees", In Proceedings of International Conference on New Methods in Language Processing, 1994.

[Hindle and Rooth] D. Hindle and M. Rooth, "Structural Ambiguity and Lexical Relations", In Proc. of the 29 th ACL, pages 229–236, 1991.

[Galbiati, 1991] G. Galbiati, "A Phrase-Based Matching Function", Journal of the American Society for Information Science, 42(1):26-48, 1991

[Allen, 1995] J. Allen, "Natural Language Understanding, Benjamin Cummings", 1995.

[TDT, 2002] Topic Detection and Tracking, URL: http://www.nist.gov/TDT, Accessed March 22, 2002.

[Donini et al., 1996] F. Donini, M. Lenzerini, D. Nardi, A. Schaerf, "Reasoning in description logics", In G. Brewka (Ed.), Principles of Knowledge Representation, pp. 191-236, CLSI Publications

[Meghini et al., 1993] C. Meghini, F. Sebastiani, U. Straccia, C. Thanos, "A Model of Infromation Retrieval Based on a

Terminological Logic“, In R. Korfhage et al (Ed.), Proceedings of SIGIR-93, pp. 298-307, ACM Press, Baltimore. 1993

[Jurafsky and Martin] Daniel Jurafsky and James H. Martin, “Speech and language processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition“, Prentice-Hall, 2000

[Litkowski, 1999] K. Litkowski, “Question-Answering Using Semantic Relation Triples“, In Proceedings of the 8th Text Retrieval Conference (TREC-8), 1999

[Denber, 1998] M. Denber, “Automatic Resolution of Anaphora in English“, Eastman Kodak Co., Imaging Science Division, 1998.

[Manning and Schütze, 2000] C. Manning and H. Schütze, “Foundations of Statistical Natural Language Processing“, MIT Press, 2000.

[Baeza-Yates and Ribeiro-Neto, 1999] R. Baeza-Yates and B. Ribeiro-Neto, “Modern Information Retrieval“, ACM Press, 1999.

[Santorini, 1990] B. Santorini, “Part-of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Rev.)“, Technical report MS-CIS-90-47, Department of Computer and Information Science, University of Pennsylvania, 1990.

[Monz and de Rijke, 2001] C. Monz and M. de Rijke, “Light-Weight Entailment Checking for Computational Semantics“, ILLC.

[Monz, 2000] C. Monz, “Computational Semantics and Information Retrieval“, In: J. Bos and M. Kohlhase (eds.) Proceedings of the 2nd Workshop on Inference in Computational Semantics (ICoS-2), 2000. pp. 1-5.

[Yang, 1998] Yiming Yang, “An Evaluation of Statistical Approaches to Text Categorization“, In Information Retrieval 1, 69-90 (1999), Kluwer 1999.

[Keenan and Comrie, 1977] E. Keenan and B. Comrie, “Noun Phrase Accessibility and Universal Grammar“, Linguistic Inquiry, 8:62-100

[Harman, 1995] D. Harman, "The Second Text Retrieval Conference (TREC-2)", in Information Processing & Management, 31(3):271-289, 1995.

[Crestani et al., 1995] F. Crestani, I. Ruthven, M. Sanderson, C.[sic!] van Rijsbergen, "The Troubles with Using a Logical Model of IR on a Large Collection of Documents" In D. Harman, (Ed.), Proceedings of the 4th Text Retrieval Conference (TREC-4), pp. 509-526, NIST Special Publication 500-236

[BLIKSEM, 2000] H. de Nivelle, Bliksem Resolution Prover, http://www.mpi-sb.mpg.de/ bliksem (Accessed July 2000)

[Zhang, 2002] J. Zhang, "Evaluating Anaphora Resolution Algorithms", http://www.cs.brandeis.edu/ jyzhang/AR/literature.html (Accessed March 2002)

[Fox, 1992] Christopher Fox, "Lexical Analysis and Stoplists" in William B. Frakes and Baeza-Yates (Eds.), "Information Retrieval: Data Structures and Algorithms", Prentice-Hall, 1992.

[SPASS, 1996] C. Weidenbach, B. Gaede, G. Rock, SPASS & FLOTTE, version 0.42, In 13th International Conference on Automated Deduction, CADE-13, LNAI, Springer

[van Rijsbergen, 1986] K. van Rijsbergen, "A non-classical logic for information retrieval", in The Computer Journal 29(6):481-485

[WordNet, 2002] WordNet, URL: http://www.cogsci.princeton.edu/ wn Accessed Junuary 15, 2002.

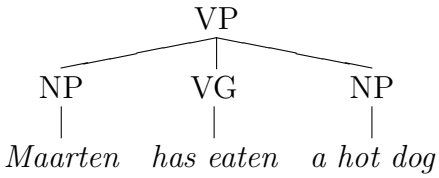
Appendix A

Appendices

A.1 Linguistic terminology used in this thesis

This is a quick description of the most frequent non-layman linguistic terminology in this paper. The commenting to the choice of terms is by no means complete, it's just quick aid and it's English specific not general.

When analyzing a natural language sentence, where do we start? Well, we observe first that some words bind together more strongly[1]. For instance:



Because *hot* and *a* qualify 'dog' rather than 'Maarten', 'has' or 'eaten' and *has* has likewise to do with eaten and nothing else. Secondly, as we understand the sentence, there is one action, namely 'to eat' with two participants. By convention we call the group of words that denote an action and specifications to it **VG** (verb group) and potential

[1]This is an observation and not an axiom, and there are obscure cases where some word of a sentence seems not to go together with any particular group more than any other

participants[2] to an action **NP**s (noun phrases) or **PP**s according as they begin with a preposition or not. **VP** (verb phrase) is then a well-formed VG with it's participants. An action can be the participant in another action etc which is expressed usually by {bf subordinate clauses or **nominalizations** (turning a verb into a noun). For instance, in "He said that he was a carpenter" we have a subordinate clause "he was a carpenter" introduced by the subordinating conjunction "that". In "Eating meat is not good" we also have an action "Eating meat" as an argument to a predicate "is" but this is a nominalization. The difference between nominalization and a subordinated clause is that the a nominalization has a **non-finite** verb and the subordinated clause has a **finite**. The finite verb causes it to be called a **clause** and normally requires a conjunction if it is to be as an argument of another action. In English, in short, the finite verbs are those marked by person and time (as in *said* which is marked for time (preterite), and *travels* is for both person (3rd) and time (present)) and the non-finite ones are the rest. The taxonomy stems from the observation that we get a consistent and useful description if we say that a clause can have one and only one finite verb (otherwise it's several clauses in some relation to each other) and has to have a finite verb (explicit or implicit) to be called a clause. Also if there is a well-formed action with a finite verb, it constitutes a clause.

As noted above, almost all actions, as we understand them, have a number of mandatory participants. "Walk" requires one, namely the one who walks, "bake" requires two, the one who bakes and what he bakes[3] and there are those that require more or are variable. The **subject** is the participant the action is syntactically congruent[4] with. The **direct object**, if there is one, is the 'second'[5] syntactical participants of the action. **Agent** is the *semantic* "doer" of the action and **patient** is

[2]According to the nature of each action, it's understood to take a certain number of participants each of which must be present in thought although not necessarily explicitly stated in words (for example if it's not in focus or it's understood from the context)

[3]If we just say "I bake" we believe there's still something being baked even though it's not mentioned

[4]In English, this is for example agreement, word order and, on pronouns, subject forms

[5]Second as in for instance importance, that which is least possible to remove to still understand the meaning of the action. There are many subdivisions of objects which won't be trated here, likewise its relations to adverbials

the *semantic* goal of the action. Subject/agent and dir. object/patient mostly coincide in English. The most common exception is passive clauses such as "The man was killed by the thieves" where "the man" is the (syntactical) subject yet he is the one being killed so "the man" is the patient. The doer:s of the action kill is undoubtedly the thieves so they're the agent.

A.2 Metaphysics of the concept of entailment relation

It might look like we are on one hand saying entailment is a pure yes/no-decision, and that we are saying this yes/no decision should be based on some variable measure like 'contains almost all the information of' or 'contain the substantial information of'. Which way should it be or is it better described as a number in [0..1]?

First we note that, even if we are more precise and choose a border by some phrase i.e 'contains more than 75,69% of the non-detail (i.e focus etc) information of', and try to adhere to it there will be problems. For any definition like this, it is not difficult to construct two NL paragraphs were we are unsure as to whether one entails the other, for instance using words of cultural or personal values or just plain vagueness. But fictious cases aside, consider the following two paragraphs (of the same article) taken from the newspaper this morning[6]:
"The explosions in the church is a setback for the military leader of Pakistan Pervez Musharraf. The attack shows that the president does not have control over the religiously coloured violence that he two months ago promised to extinguish."
"According to western diplomats in Islamabad President Musharraf has not been able to follow up the rhetorics by action."
Let's do some rough reasoning, it might seen at first that the former entails the latter, since the former says that (1) Musharraf has made that promise, and yet (2) has not prevented the violence, which entails (1) has done rhetorics but (2) has not taken action. But on second thoughts, the latter paragraph also has the *according to western diplo-*

[6]Dagens Nyheter, Monday the 18th of March, part A, p. 14

mats to it so it's either saying a different thing altogether, as in X did Y is not true iff Z says X did Y, or has extra information that is not present in the former paragraph. However, someone very deep might hold that the use of the word *setback* in the former paragraph implies some kind of western view of it anyway.

It is way beyond computers today do reasoning similar to this, therefore this section has little relevance to the computational perspective of entailment. The computation of entailment as in this thesis, outputs a number between 0 and 1, the question is whether this number says anything on the entailment concept. This relies on how we view NL entailment, for which we have discussed the limitations of certain natural characterizations. In this thesis we have chosen one, not thinking it ultimately defines the metaphysical nature of entailment but rather for practical reasons and that other characterizations suffer from uncertainties of the same degree. The computations are independent and are open to be reinterpreted for anyone with another scale.

A.3 Stopword list

A Stop List for general text from chapter 7 of [Fox, 1992].

a about above across after

again against all almost alone

along already also although always

among an and another any

anybody anyone anything anywhere are

area areas around as ask

asked asking asks at away

b back backed backing backs

be because become becomes became

been before began behind being

beings best better between big

both but by c came

can cannot case cases certain
certainly clear clearly come could
d did differ different differently
do does done down downed
downing downs during e each
early either end ended ending
ends enough even evenly ever
every everybody everyone everything everywhere
f face faces fact facts
far felt few find finds
first for four from full
fully further furthered furthering furthers
g gave general generally get
gets give given gives go
going good goods got great
greater greatest group grouped grouping
groups h had has have
having he her herself here
high higher highest him himself
his how however i if
important in interest interested interesting
interests into is it its
itself j just k keep
keeps kind knew know known
knows l large largely last
later latest least less let
lets like likely long longer
longest m made make making

man many may me member
members men might more most
mostly mr mrs much must
my myself n necessary need
needed needing needs never new
newer newest next no non
not nobody noone nothing now
nowhere number numbered numbering numbers
o of off often old
older oldest on once one
only open opened opening opens
or order ordered ordering orders
other others our out over
p part parted parting parts
per perhaps place places point
pointed pointing points possible present
presented presenting presents problem problems
put puts q quite r
rather really right room rooms
s said same saw say
says second seconds see sees
seem seemed seeming seems several
shall she should show showed
showing shows side sides since
small smaller smallest so some
somebody someone something somewhere state
states still such sure t
take taken than that the

their them then there therefore
these they thing things think
thinks this those though thought
thoughts three through thus to
today together too took toward
turn turned turning turns two
u under until up upon
us use uses used v
very w want wanted wanting
wants was way ways we
well wells went were what
when where whether which while
who whole whose why will
with within without work worked
working works would x y
year years yet you young
younger youngest your yours z

A.4 The 42 senses of of the verb 'run'

According to [WordNet, 2002], the verb run has 42 senses (first 29 from tagged texts)

1. run – (move fast by using one's feet, with one foot off the ground at any given time)
2. run, scarper, turn tail, lam, run away, bunk, break away – (take to one's heels; cut and run)
3. run, go, pass, lead, extend – (stretch out over a distance, space, time, or scope; run or extend between two points or beyond a certain point; "Service runs all the way to Cranbury"; "His knowledge doesn't go very far"; "My memory extends back to my fourth year of life"; "The facts extend beyond a consideration of her personal assets")

4. operate, run – (direct or control; of machinery, projects, businesses, etc.)
5. run, go – (have a particular form; "the story or argument runs....", "as the saying goes...")
6. run, flow, course – (move along, of liquids; "Water flowed into the cave")
7. function, work, operate, go, run – (function properly; "The washing machine won't go unless it's plugged in")
8. range, run – (change or be different within limits; "Estimates for the losses in the earthquake range as high as $2 billion"; "Interest rates run from 5 to 10 percent"; "The instruments ranged from tuba to cymbals"; My students range from very bright to dull")
9. campaign, run – (run or stand for office)
10. play, run – (cause to be played: "They ran the tapes over and over again")
11. run – (move about freely and without restraint, or act as if running around in an uncontrolled way; "who are these people running around in the building?" "She runs around telling everyone of her troubles")
12. tend, lean, incline, run – (have a tendency or disposition to do or be something; be inclined; "She tends to be nervous before her lectures")
13. run – (be running or functioning, as of engines or machines; "Is the computer running?")
14. run – (change from one state to another; "run amok"; "run rogue", "run riot")
15. run – (cause to perform; "run a subject"; "run a process")
16. run – (be affected by; be subjected to; as in "run a temperature," "run a risk")
17. prevail, persist, die hard, run, endure – (cease to exist after resistance or a struggle; "These stories die hard")
18. run – (occur persistently; "Musical talent runs in the family")
19. run – (execute a program or process, as on a computer or a machine; "Run the dishwasher"; "run a new program on the Mac")
20. carry, run – (include as the content; broadcast or publicize; "We ran the ad three times"; "This paper carries a restaurant review"; "All major networks carried the press conference")
21. run – (carry out; "run an errand")
22. guide, run, pass – (guide or pass over something; "He ran his eyes over her naked body." "She ran her fingers along the carved figurine.")

23. run, lead – (cause something to pass or lead somewhere; "Run the wire behind the cabinet")
24. run – (make without a miss; in sports or games)
25. run, black market – (deal in illegally, such as arms or liquor)
26. run – (cause an animal to move fast)
27. run, bleed – (be diffused; of dyes and colors)
28. run – (sail before the wind)
29. run – (cover by running; run a certain distance; "She ran 10 miles that day")
30. run, run for – (extend or continue for a certain period of time; "The film runs 5 hours")
31. run – (set animals loose to graze)
32. run, consort – (keep company; of male animals)
33. run – (run with the ball; in football)
34. ply, run – (travel a route regularly; "Ships ply the waters near the coast")
35. force, run, drive, ram – (physical or metaphorical, as in "She rammed her mind into focus")
36. hunt, run, hunt down, track down – (hunt wild animals; "Goering often hunted wild boars in Poland")
37. race, run – (compete in a race, as in athletics)
38. run – (be in the running; compete for a certain position; "Who's running this year?")
39. move, go, run – (progress by being changed: "The speech has to go through several more drafts"; "run through your presentation before the meeting")
40. melt, run, melt down – (reduce from a solid to a liquid state, usually by heating; "melt butter"; "melt down gold")
41. ladder, run – (come unraveled or undone as if by snagging, of stockings; "Her nylons were running")
42. run, unravel – (become undone, as of clothes such as knitted fabrics; "the sweater unraveled")

A.5 The syntactic filter of Lappin & Leass' pronoun anaphora resolution algorithm

Verbatim from [Lappin and Leass, 1994] p.3.

A.5.1 2.1.1 The Syntactic filter on PronounNP Coreference

The filter consists of six conditions for NPpronoun noncoreference within a sentence. To state these conditions, we use the following terminology. The agreement features of an NP are its number, person, and gender features. We will say that a phrase P is in the argument domain of a phrase N iff P and N are both arguments of the same head. We will say that P is in the adjunct domain of N iff N is an argument of a head H, P is the object of a preposition PREP, and PREP is an adjunct of H. P is in the NP domain of N iff N is the determiner of a noun Q and (i) P is an argument of Q, or (ii) P is the object of a preposition PREP and PREP is an adjunct of Q. A phrase P is contained in a phrase Q iff 1) P is either an argument or an adjunct of Q, i.e. P is immediately contained in Q, or 2) P is immediately contained in some phrase R, and R is contained in Q.

A pronoun P is noncoreferential with a (nonreflexive or nonreciprocal) noun phrase N if any of the following conditions hold:

(a) P and N have incompatible agreement features.

(b) P is in the argument domain of N.

(c) P is in the adjunct domain of N.

(d) P is an argument of a head H, N is not a pronoun, and N is contained in H.

(e) P is in the NP domain of N.

(f) P is a determiner of a noun Q, and N is contained in Q.

Examples of coindexings that would be rejected by these conditions are given in figure 1.

Condition 1: The $woman_i$ said that he_j is funny.

Condition 2: She_i likes her_j. $John_i$ seems to want to see him_j.

Condition 3: She_i sat near her_j.

Condition 4: He_i believes that the man_j is amusing. This is the man_i he_j said $John_j$ wrote about.

Condition 5: $John_i$'s portrait of him_j is interesting.

Condition 6: His_i portrait of $John_j$ is interesting. His_i description of the portrait by $John_j$ is interesting.

A.6 List of Tags

This description is taken from [Santorini, 1990] where more examples and discussions of specific cases can be found.

```
Tag
 Description
 Examples

\$
 dollar
 $ -$ --$ A$ C$ HK$ M$ NZ$ S$ U.S.$ US$

‘‘
 opening quotation mark
 ‘ ‘‘

’’
 closing quotation mark
 ’ ’’

(
 opening parenthesis
 ( [ {

)
 closing parenthesis
```

```
 ) ] }

,
 comma
 ,

--
 dash
 --

.
 sentence terminator
 . ! ?

:
 colon or ellipsis
 : ; ...

CC
 conjunction, coordinating
 \& 'n and both but either et for less minus neither nor or plus
 so therefore times v. versus vs. whether yet

CD
 numeral, cardinal
 mid-1890 nine-thirty forty-two one-tenth ten million 0.5 one
 forty-seven 1987 twenty '79 zero two 78-degrees eighty-four IX
 '60s .025 fifteen 271,124 dozen quintillion DM2,000 ...

DT
 determiner
 all an another any both del each either every half la many much
 nary neither no some such that the them these this those

EX
 existential there
 there
```

```
FW
 foreign word
 gemeinschaft hund ich jeux habeas Haementeria Herr K'ang-si vous
 lutihaw alai je jour objets salutaris fille quibusdam pas trop
 Monte terram fiche oui corporis ...

IN
 preposition or conjunction, subordinating
 astride among uppon whether out inside pro despite on by
 throughout below within for towards near behind atop around if
 like until below next into if beside ...

JJ
 adjective or numeral, ordinal
 third ill-mannered pre-war regrettable oiled calamitous first
 separable ectoplasmic battery-powered participatory fourth
 still-to-be-named multilingual multi-disciplinary ...

JJR
 adjective, comparative
 bleaker braver breezier briefer brighter brisker broader bumper
 busier calmer cheaper choosier cleaner clearer closer colder
 commoner costlier cozier creamier crunchier cuter ...

JJS
 adjective, superlative
 calmest cheapest choicest classiest cleanest clearest closest
 commonest corniest costliest crassest creepiest crudest cutest
 darkest deadliest dearest deepest densest dinkiest ...

LS
 list item marker
 A A. B B. C C. D E F First G H I J K One SP-44001 SP-44002
 SP-44005 SP-44007 Second Third Three Two \* a b c d first five
 four one six three two

MD
 modal auxiliary
```

```
 can cannot could couldn't dare may might must need ought shall
 should shouldn't will would

NN
 noun, common, singular or mass
 common-carrier cabbage knuckle-duster Casino afghan shed
 thermostat investment slide humour falloff slick wind hyena
 override subhumanity machinist ...

NNP
 noun, proper, singular
 Motown Venneboerger Czestochwa Ranzer Conchita Trumplane
 Christos Oceanside Escobar Kreisler Sawyer Cougar Yvette Ervin
 ODI Darryl CTCA Shannon A.K.C. Meltex Liverpool ...

NNPS
 noun, proper, plural
 Americans Americas Amharas Amityvilles Amusements
 Anarcho-Syndicalists Andalusians Andes Andruses Angels Animals
 Anthony Antilles Antiques Apache Apaches Apocrypha ...

NNS
 noun, common, plural
 undergraduates scotches bric-a-brac products bodyguards facets
 coasts divestitures storehouses designs clubs fragrances
 averages subjectivists apprehensions muses factory-jobs ...

PDT
 pre-determiner
 all both half many quite such sure this

POS
 genitive marker
 ' 's

PRP
 pronoun, personal
 hers herself him himself hisself it itself me myself one oneself
```

```
 ours ourselves ownself self she thee theirs them themselves they
 thou thy us

PRP\$
 pronoun, possessive
 her his mine my our ours their thy your

RB
 adverb
 occasionally unabatingly maddeningly adventurously professedly
 stirringly prominently technologically magisterially
 predominately swiftly fiscally pitilessly ...

RBR
 adverb, comparative
 further gloomier grander graver greater grimmer harder harsher
 healthier heavier higher however larger later leaner lengthier
 less-perfectly lesser lonelier longer louder lower more ...

RBS
 adverb, superlative
 best biggest bluntest earliest farthest first furthest hardest
 heartiest highest largest least less most nearest second
 tightest worst

RP
 particle
 aboard about across along apart around aside at away back before
 behind by crop down ever fast for forth from go high i.e. in into
 just later low more off on open out over per pie raising start
 teeth that through under unto up up-pp upon whole with you

SYM
 symbol
 % & ' '' ''. ) ). * + ,. < = > @ A[fj] U.S U.S.S.R \* \*\* \*\*\*

TO
 "to" as preposition or infinitive marker
```

```
 to

UH
 interjection
 Goodbye Goody Gosh Wow Jeepers Jee-sus Hubba Hey Kee-reist Oops
 amen huh howdy uh dammit whammo shucks heck anyways whodunnit
 honey golly man baby diddle hush sonuvabitch ...

VB
 verb, base form
 ask assemble assess assign assume atone attention avoid bake
 balkanize bank begin behold believe bend benefit bevel beware
 bless boil bomb boost brace break bring broil brush build ...

VBD
 verb, past tense
 dipped pleaded swiped regummed soaked tidied convened halted
 registered cushioned exacted snubbed strode aimed adopted belied
 figgered speculated wore appreciated contemplated ...

VBG
 verb, present participle or gerund
 telegraphing stirring focusing angering judging stalling
 lactating hankerin' alleging veering capping approaching
 traveling besieging encrypting interrupting erasing wincing ...

VBN
 verb, past participle
 multihulled dilapidated aerosolized chaired languished panelized
 used experimented flourished imitated reunified factored
 condensed sheared unsettled primed dubbed desired ...

VBP
 verb, present tense, not 3rd person singular
 predominate wrap resort sue twist spill cure lengthen brush
 terminate appear tend stray glisten obtain comprise detest
 tease attract emphasize mold postpone sever return wag ...
```

```
VBZ
 verb, present tense, 3rd person singular
 bases reconstructs marks mixes displeases seals carps weaves
 snatches slumps stretches authorizes smolders pictures emerges
 stockpiles seduces fizzes uses bolsters slaps speaks pleads ...

WDT
 WH-determiner
 that what whatever which whichever

WP
 WH-pronoun
 that what whatever whatsoever which who whom whosoever

WP\$
 WH-pronoun, possessive
 whose

WRB
 Wh-adverb
 how however whence whenever where whereby whereever wherein
 whereof why
```